THE HAWAII GEOTHERMAL PROJECT

THE EFFECT OF DIKE INTRUSION ON FREE CONVECTION IN GEOTHERMAL RESERVOIRS

TECHNICAL REPORT NO. 7

HAWAII GEOTHERMAL PROJECT

ENGINEERING PROGRAM

THE EFFECT OF DIKE INTRUSION ON FREE CONVECTION IN GEOTHERMAL RESERVOIRS

TECHNICAL REPORT No. 7

December 1, 1974

Prepared Under

NATIONAL SCIENCE FOUNDATION

RESEARCH GRANT NO. GI-38319

By

K. H. Lau

and

Ping Cheng

Hilo College
University of Hawaii
Hilo, Hawaii 96720

College of Engineering
University of Hawaii
Honolulu, Hawaii 96822

TABLE OF CONTENTS

ABSTRACT

A perturbation analysis is made for the simultaneous heat and mass transfer in unconfined geothermal reservoirs with dike intrusion. The perturbation equations are of elliptic type that can be solved numerically by the finite difference method. Up to the second-order approximations are retained in the numerical computation. The effects of dike intrusion on streamlines, temperature distribution, and the shape of water table in two-dimensional aquifers with low permeability are shown.

LIST OF SYMBOLS

D	$D \equiv \rho_s K g h / \alpha \mu$
F	function denoting the position of water table
g	gravity vector
h	depth of the reservoir at the ocean sides
k_m	thermal conductivity of the porous medium
K	permeability of the porous medium
ℓ	the width of the reservoir
L	the dimensionless width of the reservoir, $L \equiv \ell/h$
m	dummy index in Eq. (10)
$\bar{n}$	unit vector normal to water table
p	pressure
P	dimensionless pressure, $P \equiv (p-p_a) / \rho_s g h$
P_0, P_1	zero-order and first-order perturbation functions for pressure
T	temperature
T_d	temperature of the dike
T_L	prescribed temperature of the impermeable surface
u, v	velocity components in the x and y directions
x, y	Cartesian coordinate system
X, Y	dimensionless coordinates
α	equivalent thermal diffusivity, $\alpha \equiv k_m/(\rho C_p)_f$
β	thermal expansion coefficient
ε	perturbation parameter, $\varepsilon \equiv \beta (T_c-T_s)$
η	the height of water table
$\bar{\eta}$	dimensionless height of water table, $\bar{\eta} \equiv \eta/h$

η_1, η_2	zero-order and first-order perturbation functions for the height of water table
θ	dimensionless temperature, $\theta \equiv (T-T_s) / (T_c-T_s)$
θ_0, θ_1, θ_2	zero-order, first-order and second-order perturbation functions for temperature
θ_d	prescribed dimensionless temperature of the dike
θ_L	prescribed dimensionless temperature of the impermeable surface
μ	viscosity of convecting fluid
ρ	density of convecting fluid
ψ	stream function
Ψ	dimensionless stream function, $\Psi \equiv \mu\psi/\rho_s ghK$
Ψ_1, Ψ_2	first-order and second-order dimensionless stream functions

Subscripts

a	atmospheric condition
s	condition in the ocean

LIST OF FIGURES

INTRODUCTION

Magmatic intrusion occurs frequently in the shallow parts of the earth's crust in some parts of the world where there are intense tectonic activities. The intruded magma then acts as a heat source which in turn heats the groundwater in the aquifer directly or indirectly. The heated groundwater is driven buoyantly upward to the top of the aquifer where it can then be tapped for power generation through drill holes. A qualitative assessment of the capacity and location of geothermal resources can sometimes be made from the observation of temperature anomaly in a rock formation or heat flux anomaly on the earth's surface. A thorough understanding of heat transfer characteristics in the earth's crust thus will aid in a correct interpretation of field data during geophysical exploration.

The intrusive magma may take many different forms or sizes. A sheet-like intrusive body is called a dike or a sill depending upon whether it is perpendicular or parallel to the stratification in the bedded rocks. On the basis of the heat conduction theory, Horai [1] has recently completed a study to relate surface heat flux to the parameters specifying the intrusion such as magmatic temperature, geometry, and dimensions of the intrusive body. However, recent studies [2-4] suggest that the convection of ground water plays an important role on the heat transfer characteristics in geothermal areas. Thus, in the present paper, we shall study the effects of dike intrusion on the temperature distribution in an island aquifer (Fig. 1A) taking into account the movement of ground water. To simplify the problem, the dike is idealized as a vertical impermeable rectangular obstacle while the island aquifer is idealized as a two-dimensional homogeneous and isotropic porous medium bounded vertically by ocean on the sides, with horizontal impermeable surface at the bottom, and unconfined at the top where the position of water table is not known apriori (Fig. 1B). The governing non-linear partial differential equations for the simultaneous heat and mass transfer in the porous medium are approximated by a set of linear subproblems on the basis of a perturbation method [2] which is applicable for reservoirs with low permeability. Up to the second-order approximations are retained in the numerical computation. Contours for streamlines and temperature distribution, as well as the amount of the upwelling of water table as a result of dike intrusion for a particular set of parameters are presented.

Governing Equations and Boundary Conditions

The governing equations for the simultaneous heat and mass transfer in a porous medium are the continuity equation, Darcy's law, energy equation, and equation of state. To simplify the formulation of the problem, we assume that:

1. The flow field is steady and two-dimensional.
2. There is no rainfall at the water table.
3. The temperature of the fluid, T, is everywhere below boiling for the pressure, p, at that depth.
4. The fluid properties such as specific heat, C, and the kinematic viscosity, μ, as well as the medium properties such as thermal conductivity, km, and permeability, k, are all constant.
5. Density, ρ, is linearly proportional to temperature, i.e., $\rho = \rho_s\,[1-\beta(T-T_s)]$ where β is the thermal expansion coefficient and the subscript "s" denoting the condition in the ocean.
6. Boussinesq approximation is employed, i.e., density is assumed to be constant except in the bouyancy force term.

With these assumptions it can be shown that the governing equations in terms of dimensionless pressure, P, and temperature, θ, are [2]

$$\frac{\partial^2 P}{\partial X^2} + \frac{\partial^2 P}{\partial Y^2} = \varepsilon \frac{\partial \theta}{\partial Y} \quad , \tag{1}$$

$$\frac{\partial^2 \theta}{\partial X^2} + \frac{\partial^2 \theta}{\partial Y^2} + D \left\{ \left[\frac{\partial P}{\partial X} \frac{\partial \theta}{\partial X} + \frac{\partial P}{\partial Y} \frac{\partial \theta}{\partial Y} \right] + [1 - \varepsilon\theta] \frac{\partial \theta}{\partial Y} \right\} = 0 \quad . \tag{2}$$

where

$$P \equiv \frac{p - p_a}{\rho_s g h}, \quad \theta \equiv \frac{T-T_s}{T_c-T_s}, \quad X \equiv \frac{x}{h}, \quad Y \equiv \frac{y}{h}, \quad L \equiv \frac{\ell}{h},$$

$$\varepsilon \equiv \beta(T_c-T_s) \text{ , and } D \equiv \frac{\rho_s k g h}{\alpha \mu} \quad , \tag{3}$$

with h and ℓ denoting the depth and the width of the aquifer, g the gravitational acceleration, and $\alpha \equiv \frac{km}{(\rho C)_f}$ the equivalent thermal diffusivity. The subscript "c" denotes some reference temperature.

The dimensionless boundary conditions along the ocean are

$$P\ (0,\ Y) = 1 - Y \quad , \tag{4a}$$

$$P\ (L,\ Y) = 1 - Y \quad , \tag{4b}$$

$$\theta\ (0,\ Y) = 0 \quad , \tag{4c}$$

$$\theta\ (L,\ Y) = 0 \quad , \tag{4d}$$

where Eqs. (4a) and (4b) denote hydrostatic pressure and (4c) and (4d) denote constant temperature along the ocean sides.

Along the water table $Y = \bar{\eta}$, the dimensionless boundary conditions are given by

$$P\ (X,\ \bar{\eta}) = 0 \quad , \tag{5a}$$

$$\theta\ (X,\ \bar{\eta}) = \theta_a \quad , \tag{5b}$$

$$\frac{\partial \bar{\eta}}{\partial X} \frac{\partial P}{\partial X}\ (X,\ \bar{\eta}) - \left(\frac{\partial P}{\partial Y}\ (X,\ \bar{\eta}) + 1 - \varepsilon\theta_a \right) = 0 \quad , \tag{5c}$$

where $\bar{\eta} \equiv \frac{\eta}{h}$, and $\theta_a \equiv (T_a - T_s)/(T_c - T_s)$ with T_a denoting the atmospheric temperature. It is worth mentioning that boundary condition (5c) follows from the conditions $\bar{v} \cdot \bar{n} = 0$ where $\bar{n}$ is the unit vector normal to the water table which is given by $\bar{n} = \nabla F / |\nabla F|$ with $F(X,Y) = Y - \bar{\eta}(X) = 0$ denoting the equation for the position of the water table [2].

If a vertical impermeable dike with uniform temperature T_d exists (Fig. 1B), the dimensionless boundary conditions along the dike are

$$\theta\ (X_1,\ Y) = \theta\ (X_2, Y) = \theta_d \quad , \qquad 0 \leq Y \leq Y_1 \tag{6a}$$

$$\theta\ (X,\ Y_1) = \theta_d \quad , \qquad X_1 \leq X \leq X_2 \tag{6b}$$

$$\frac{\partial P}{\partial X}\ (X_1,\ Y) = \frac{\partial P}{\partial X}\ (X_2,\ Y) = 0 \quad , \qquad 0 \leq Y \leq Y_1 \tag{6c}$$

$$\frac{\partial P}{\partial X}\ (X,\ Y_1) = -1 + \varepsilon\theta_d \quad , \qquad X_1 \leq X \leq X_2 \tag{6d}$$

where $\theta_d \equiv (T_d-T_s)/(T_c-T_s)$. It should be noted that Eqs. (6c) and (6d) follow from Darcy's law and the fact that velocity normal to the surface of the dike vanishes. Similarly, if temperature on the rest of the horizontal impermeable surface is prescribed, the boundary conditions are given by

$$\theta\ (X,\ 0) = \theta_L(X) \quad , \qquad 0 \leq X \leq X_1 \quad \text{or} \quad X_2 \leq X \leq L \tag{7a}$$

$$\frac{\partial P}{\partial Y}\ (X,\ 0) = -1 + \varepsilon\theta_L(X) \quad , \qquad 0 \leq X \leq X_1 \quad \text{or} \quad X_2 \leq X \leq L \quad . \tag{7b}$$

Eqs. (1) and (2) with boundary conditions (4)-(7) are a set of non-linear partial differential equations with non-linear boundary conditions for the determination of pressure and temperature in a hot-water aquifer with a vertical dike.

After θ is obtained, the dimensionless stream function $\psi \equiv \frac{\mu\Psi}{\rho_s ghK}$ can be determined from

$$\frac{\partial^2\psi}{\partial X^2} + \frac{\partial^2\psi}{\partial Y^2} = -\varepsilon\frac{\partial\theta}{\partial X} \quad , \tag{8}$$

where Eq. (8) is obtained from the elimination of pressure in Darcy's law and from the definition of the stream function, i.e., $u = \frac{\partial\Psi}{\partial y}$ and $v = -\frac{\partial\Psi}{\partial x}$. The boundary conditions for ψ along the ocean are given by

$$\frac{\partial\psi}{\partial X}\ (0,\ Y) = \frac{\partial\psi}{\partial X}\ (L,\ Y) = 0 \quad , \tag{9a}$$

i.e., the vertical velocity is zero along the ocean implying that $\theta_L(X) = 0$ as $X \to 0$ & $X \to L$. Along the water table, the impermeable surface, and on the dike, the boundary condition for ψ is

$$\psi = 0. \tag{9b}$$

Perturbation Analysis

Since the value of ε in Eqs. (6)-(8) is small, we shall now obtain a perturbation solution to the problem. For this purpose, we now assume that dependent variables be expanded in a power series of ε. Thus, we have

$$\theta\ (X,\ Y) = \sum_{m=0}^{\infty} \varepsilon^m \theta_m (X,\ Y) \quad , \tag{10a}$$

$$P\ (X,\ Y) = (1-Y) + \sum_{m=1}^{\infty} \varepsilon^m P_m (X,\ Y) \quad , \tag{10b}$$

$$\psi\ (X,\ Y) = \sum_{m=1}^{\infty} \varepsilon^m \psi_m (X,\ Y) \quad , \tag{10c}$$

$$\overline{\eta}(X) = 1 + \sum_{m=1}^{\infty} \varepsilon^m \eta_m (X) \quad , \tag{10d}$$

where $P_m(X,Y)$, $\theta_m(X,Y)$, $\psi_m(X,Y)$ and $\eta_m(X)$ are perturbation functions to be determined. Substituting Eq. (10) into Eqs. (1)-(9), making a Taylor's series expansion on boundary conditions (5) and (9c), and collecting terms of like power in ε, we have the following set of subproblems.

Zero-Order Approximations

The zero-order problem for θ is given by

$$\frac{\partial^2 \theta_0}{\partial X^2} + \frac{\partial^2 \theta_0}{\partial Y^2} = 0 \quad , \tag{11}$$

with boundary conditions given by

$$\theta_0\ (0,\ Y) = \theta_0\ (L,\ Y) = 0 \quad , \tag{12a}$$

$$\theta_0\ (X,\ 1) = \theta_a \quad , \tag{12b}$$

$$\theta_0\ (X,\ 0) = \theta_L\ (X) \quad , \tag{12c}$$

$$\theta_0 (X_1, Y) = \theta_0 (X_2, Y) = \theta_d \quad , \quad \text{for } 0 \leq Y \leq Y_1 \quad , \tag{12d}$$

$$\theta_0 (X, Y_1) = \theta_d, \quad \text{for } X_2 \leq X \leq X_1 \quad . \tag{12e}$$

First-Order Approximations

The first-order problem for P is given by

$$\frac{\partial^2 P_1}{\partial X^2} + \frac{\partial^2 P_1}{\partial Y^2} = \frac{\partial \theta_0}{\partial Y} \quad , \tag{13}$$

where the right hand side of Eq. (13) is known from the zero-order problem. The boundary conditions for P_1 are given by

$$P_1 (0, Y) = P_1 (L, Y) = 0 \quad , \tag{14a}$$

$$\frac{\partial P_1}{\partial Y} (X, 1) = \theta_a \quad , \tag{14b}$$

$$\frac{\partial P_1}{\partial Y} (X, 0) = \theta_L(X) \quad , \tag{14c}$$

$$\frac{\partial P_1}{\partial X} (X_1, Y) = \frac{\partial P_1}{\partial X} (X_2, Y) = 0 \quad , \quad \text{for } 0 \leq Y \leq Y_1 \tag{14d}$$

$$\frac{\partial P_1}{\partial Y} (X, Y_1) = \theta_d \quad , \quad \text{for } X_1 \leq X \leq X_2 . \tag{14e}$$

Once P_1 is determined, $\eta_1(X)$ is obtained from

$$\eta_1(X) = P_1 (X, 1) \quad . \tag{15}$$

The first-order approximation for θ is

$$\frac{\partial^2\theta_1}{\partial X^2} + \frac{\partial^2\theta_1}{\partial Y^2} = -D\left[\frac{\partial P_1}{\partial X}\frac{\partial\theta_0}{\partial X} + \frac{\partial P_1}{\partial Y}\frac{\partial\theta_0}{\partial Y} - \theta_0\frac{\partial\theta_0}{\partial Y}\right] \quad , \tag{16}$$

with boundary conditions given by

$$\theta_1(0, Y) = \theta_1(L, Y) = 0 \quad , \tag{17a}$$

$$\theta_1(X, 0) = 0 \quad , \tag{17b}$$

$$\theta_1(X, 1) = -P_1(X, 1)\frac{\partial\theta_0}{\partial Y}(X, 1) \quad , \tag{17c}$$

$$\theta_1(X_1, Y) = \theta_1(X_2, Y) = 0 \quad , \quad \text{for} \quad 0 \leq Y \leq Y_1 \tag{17d}$$

$$\theta_1(X, Y) = 0 \quad , \quad \text{for } X_1 \leq X \leq X_2 \quad . \tag{17e}$$

The first-order approximation for ψ is given by

$$\frac{\partial^2\psi_1}{\partial X^2} + \frac{\partial^2\psi_1}{\partial Y^2} = -\frac{\partial\theta_0}{\partial X_1} \quad . \tag{18}$$

with boundary conditions given by

$$\frac{\partial\psi_1}{\partial X}(0, Y) = \frac{\partial\psi_1}{\partial X}(L, Y) = 0 \quad , \tag{19a}$$

$$\psi_1(x, 1) = \psi_1(x, 0) = 0 \quad , \tag{19b}$$

and $\psi_1 = 0$ along the surface of the dike.

Second-Order Approximations

Second-order term for pressure is given by

$$\frac{\partial^2 P_2}{\partial X^2} + \frac{\partial^2 P_2}{\partial Y^2} = \frac{\partial \theta_1}{\partial Y} \quad , \tag{20}$$

with boundary conditions given by

$$P_2\ (0,\ Y) = P_2\ (L,\ Y) = 0 \quad , \tag{21a}$$

$$\frac{\partial P_2}{\partial Y}\ (X,\ 0) = 0 \quad , \tag{21b}$$

$$\frac{\partial P_2}{\partial Y}\ (X,\ 1) = \left[\frac{\partial P_1}{\partial X}\ (X,\ 1)\right]^2 - \frac{\partial^2 P_1}{\partial Y^2}\ (X,\ 1)\ \ P_1(X,\ 1) \quad , \tag{21c}$$

$$\frac{\partial P_2}{\partial X}\ (X_1,\ Y) = \frac{\partial P_2}{\partial X}\ (X_2,\ Y) = 0 \ , \quad 0 \leq Y \leq Y_1 \quad , \tag{21d}$$

$$\frac{\partial P_2}{\partial Y}\ (X,\ Y_1) = 0 \ , \quad X_1 \leq X \leq X_2 \quad . \tag{21e}$$

The second-order term for temperature is given by

$$\frac{\partial^2 \theta_2}{\partial X^2} + \frac{\partial^2 \theta_2}{\partial Y^2} = -D\left[\frac{\partial P_1}{\partial X}\frac{\partial \theta_1}{\partial X} + \frac{\partial P_1}{\partial Y}\frac{\partial \theta_1}{\partial Y} + \frac{\partial P_2}{\partial X}\frac{\partial \theta_o}{\partial X}\right.$$

$$\left. + \frac{\partial P_2}{\partial Y}\frac{\partial \theta_o}{\partial Y} - \theta_1\frac{\partial \theta_o}{\partial Y} - \theta_o\frac{\partial \theta_1}{\partial Y}\right] \quad , \tag{22}$$

with boundary conditions given by

$$\theta_2\ (0,\ Y) = \theta_2\ (L,\ Y) = 0 \quad , \tag{23a}$$

$$\theta_2\ (X,\ 0) = 0 \quad , \tag{23b}$$

$$\theta_2 (X, 1) = - \eta_1 \frac{\partial \theta_1}{\partial Y} (X, 1) - \eta_2 \frac{\partial \theta_o}{\partial Y} (X, 1) - \eta_1{}^2 \frac{\partial^2 \theta_o}{\partial Y^2} (X, 1) \quad , \tag{23c}$$

$$\theta_2 (X_1, Y) = \theta_2 (X_2, Y) = 0 \quad , \qquad 0 \leq Y \leq Y_1 \tag{23d}$$

$$\theta_2 (X, Y_1) = 0 \quad , \qquad X_1 \leq X \leq X_2 \tag{23e}$$

where η_1 is given by Eq. (15) and

$$\eta_2 = P_2 (X, 1) + P_1 (X, 1) \frac{\partial P_1}{\partial Y} (X, 1) \quad . \tag{23f}$$

The second-order approximation for ψ_2 is given by

$$\frac{\partial^2 \psi_2}{\partial X^2} + \frac{\partial^2 \psi_2}{\partial Y^2} = - \frac{\partial \theta_1}{\partial X} \, , \tag{24}$$

with boundary conditions given by

$$\frac{\partial \psi_2}{\partial X} (L, Y) = \frac{\partial \psi_2}{\partial X} (0, Y) = 0 \quad , \tag{25a}$$

$$\psi_2 (X, 0) = 0 \quad , \tag{25b}$$

$$\psi_2 (X, 1) = -\eta \frac{\partial \psi_1}{\partial Y} (X, 1) \quad , \tag{25c}$$

$$\psi_2 (X_1, Y) = \psi_2 (X_2, Y) = 0 \quad , \qquad 0 \leq Y \leq Y_1 \tag{25d}$$

$$\psi_2 (X, Y_1) = 0 \quad , \qquad X_1 \leq X \leq X_2 \quad . \tag{25e}$$

The governing equations for the zero-, first-, and second-order problems as given by Eqs. (11), (13), (16), (18), (20), (22), and (24) are either the Laplace equation or Poisson equation with nonhomogeneous boundary conditions, which could have been solved in closed form by a separation of variables. Since the numerical evaluation of the resultant expressions in terms of many double and triple Fourier series will be very costly, we therefore resort to the numerical solution of the problem by the finite difference method.

NUMERICAL COMPUTATION AND RESULTS

The Laplace operators in Eqs. (11), (13), (16), (18), (20), (22), and (24) are approximated by the standard five-point formula of the finite difference method, while the derivatives in the boundary conditions for these equations are approximated by the central difference scheme. Computations begin with the determination of θ_o, and in the order of P_1, θ_1, ψ_1, P_2, θ_2, and ψ_2 so that the derivatives in the nonhomogeneous terms of the equations for each subproblem is solved numerically by the Gauss-seidel iteration method. Computations were carried out up to the second-order approximation for D = 500 (upper bound for which the perturbation method is valid) with L = 4, ε = 0.1, and θ_a = 0.02 for the following five cases with different prescribed temperatures of θ_d and θ_L.

Case A. One vertical heat source. For the problem of geothermal heating due to a hot dike 0.5 unit in height and 2 units in width located at the center of the aquifer with a cold impermeable surface at the bottom, the prescribed temperatures are

$$\theta_d = 1 \quad , \qquad 1.9 \leq X \leq 2.1 \text{ and } 0 \leq Y \leq 0.5$$

$$\theta_L(X) = 0 \quad , \qquad 0 \leq X \leq 1.9 \text{ and } 2.1 \leq X \leq 4.$$

Case B. One horizontal heat source. For comparison, computations were also carried out for a geothermal reservoir without a dike, but with geothermal heating from the bottom impermeable surface having the following prescribed temperature

$$\theta_L(X) = \exp\left[-\left(\frac{X-2}{0.5}\right)^2\right].$$

Case C. Combined vertical and horizontal heat sources. Heating in this case is due to the combination of a vertical dike located at the center of the reservoir as in Case A, and the hot impermeable surface as in Case B. The prescribed temperatures for the heat sources are

$$\theta_d = 1 \quad , \qquad 1.9 \leq X \leq 2.1 \text{ and } 0 \leq Y \leq 0.5$$

$$\theta_L(X) = \exp\left[-\left(\frac{X-2}{0.5}\right)^2\right] \quad , \qquad 0 \leq X \leq 1.9 \text{ and } 2.1 \leq X \leq 4.0$$

Case D. Two vertical heat sources. Computations were also carried out for two hot dikes at the same temperature with a cold impermeable bottom surface. The two dikes are of the same thickness (2 units in width) but with different heights where the left dike (at $1.4 \leq X \leq 1.5$) is 0.6 unit in height while the right dike (at $2.5 \leq X \leq 2.6$) is 0.3 unit in height. The prescribed temperatures are

$$\theta_d = 1 \text{ , for } \quad 1.4 \leq X \leq 1.5 \quad \text{and} \quad 0 \leq Y \leq 0.6$$

$$\theta_d = 1 \text{ , for } \quad 2.5 \leq X \leq 2.6 \quad \text{and} \quad 0 \leq Y \leq 0.3$$

with $\theta_L = 0$ elsewhere on the bottom impermeable surface.

Case E. Two horizontal heat sources. Computations for this case were carried out so that results can be compared with Case D. There are no dikes existing in Case E and the heating is due to the hot bottom impermeable surface with two relatively maximum temperatures located at X = 1.45 and X = 2.55 corresponding to the locations of the dikes in Case D. The prescribed temperatures are

$$\theta_L = \exp\left[-\left(\frac{X-1.45}{0.5}\right)^2\right] \quad , \qquad X \leq 2.0$$

$$\theta_L = \exp\left[-\left(\frac{X-2.55}{0.5}\right)^2\right] \quad , \qquad X > 2.0 \ .$$

Results of Cases A, B, and C are plotted in Figs. 2-6. Both the flow pattern and temperature contours are symmetric with respect to X = 2. For clarity, however, only $\psi = 0.001$ and $\psi = 0.0004$ are plotted in either side of the aquifer as shown in Fig. 2. The streamlines for the three cases exhibit similar behavior with cold water moving inland in the lower portion of the island aquifer and warm water discharging into the ocean in the upper portion of the aquifer. Near the heat sources, a column of hot fluid rises rapidly which induces two convective cells on either side of the heat sources. When a hot dike exists in the reservoir, the heat source becomes relatively close to the water table. This, plus the fact that the dike provides a larger area for heat transfer, makes it a possibility that hot water can be found at shallow depths, as is shown in Fig. 3. The horizontal temperature distributions at

Y = 0.2 and Y = 0.4 for Cases A, B, and C are plotted in Fig. 4, where it is shown that the rate of change in temperature is rapid at the region near the heat source. This boundary layer behavior in temperature distribution is most pronounced for Case A. It is of interest to note that the vertical velocity distribution (Fig. 5) has the same shape as the temperature distribution (Fig. 4), and that "velocity slip" occurs on the impermeable surface. The effects of vertical and horizontal heating on the amount of upwelling of water table are shown in Fig. 6 -- it can be seen here that upwelling of water table increases if a hot dike exists.

Results of Cases D and E are plotted in Figs. 7-9. The streamlines for Cases D and E are shown in Figs. 7a and 7b, where it is noted that the convective pattern in the region left of X = 1.55 (i.e., at the point of maximum heating) for both cases are similar qualitatively. In both of these cases, a column of hot fluid near X = 1.55 rises all the way up to the top of the water table and then moves either to the right or left and discharges into the ocean. Consequently, cold water entering from the left of the aquifer does not mix with those from the right, and vise versa. A comparison of Figs. 7a and 7b shows that the convective pattern in the region to the right of X = 1.55 is different for Cases D and E. Of particular interest is the region between the two dikes in Fig. 7a. It is noted that, although the dikes are only 0.6 and 0.3 units in height, they act as a complete barrier for the movement of ground water, thus sealing off the seepage from the ocean. On the other hand, without a dike at X = 2.55, groundwater moves freely as is shown in Fig. 7b. Figs. 8 and 9 show the temperature contours and the upwelling of water table for Cases D and E. Again, it is shown that warm water at shallow depth is possible and that the amount of upwelling of water table increases, whenever there is a dike.

Concluding Remarks

The present perturbation method is valid only for small ε as well as for small and moderate values of D which is applicable to reservoirs with low permeability. For extremely large values of D, the solution will break down in the region near the heat source as well as directly above the heat source where convection is predominant. Thus for reservoirs with high permeability the method of matched asymptotic solution (5) must be used. The senior author (P. Cheng) is presently working on this singular perturbation problem.

ACKNOWLEDGMENT

The authors wish to thank Professors G. Keller and A. Furumoto for suggesting the problem during the Conference on the Utilization of Volcano Energy held in Hilo, Hawaii on February 4-8, 1974.

REFERENCES

1. Horai, K., "Heat Flow Anomaly Associated with Dike Intrusion," *J. Geophys. Res.*, Vol. 79, 1974, pp. 1640-1646.

2. Cheng, P. and K.H. Lau, "Steady State Free Convection in an Unconfined Geothermal Reservoir," *Hawaii Geothermal Project, Technical Report No. 2*, College of Engineering, University of Hawaii, Honolulu, Hawaii, March 1, 1974. Accepted for publication in the *J. Geophys. Res.*

3. Domenico, P.A. and Palciauskas, "Theoretical Analysis of Forced Convective Heat Transfer in Regional Ground-Water Flow," *Geological Society of America Bulletin*, Vol. 84, 1973, pp. 3803-3814.

4. Henry, H.R. and F.A. Kohout, "Circulation Patterns of Saline Groundwater Affected by Geothermal Heating--as Related to Waste Disposal," *Underground Waste Management and Environmental Implications*, Vol. 18, 1973, pp. 202-221.

5. Van Dyke, M., *Perturbation Methods in Fluid Mechanics*, Academic Press, New York, 1964.

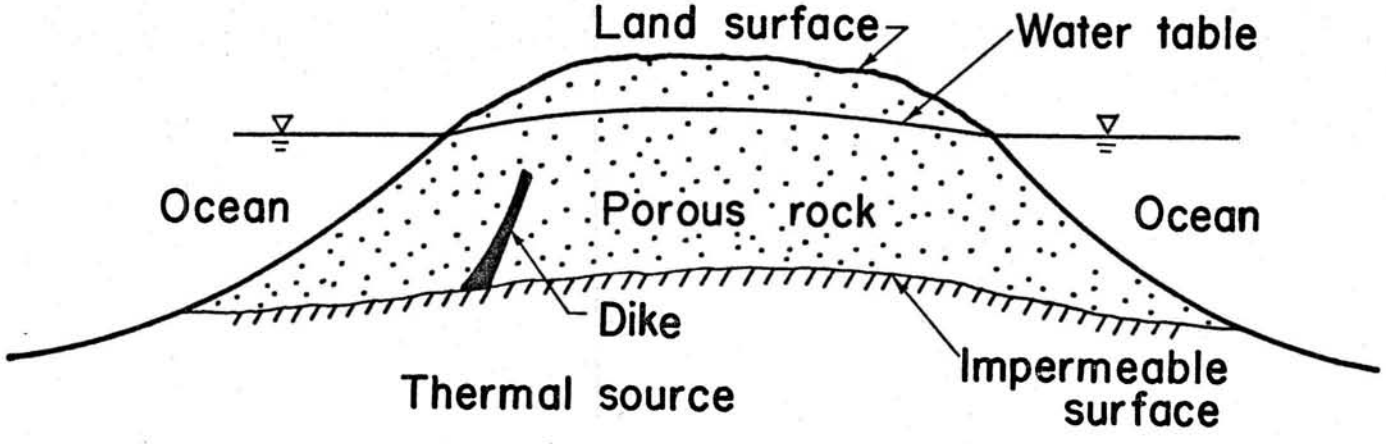

Fig. 1A AN UNCONFINED AQUIFER IN A VOLCANIC ISLAND WITH DIKE INTRUSION

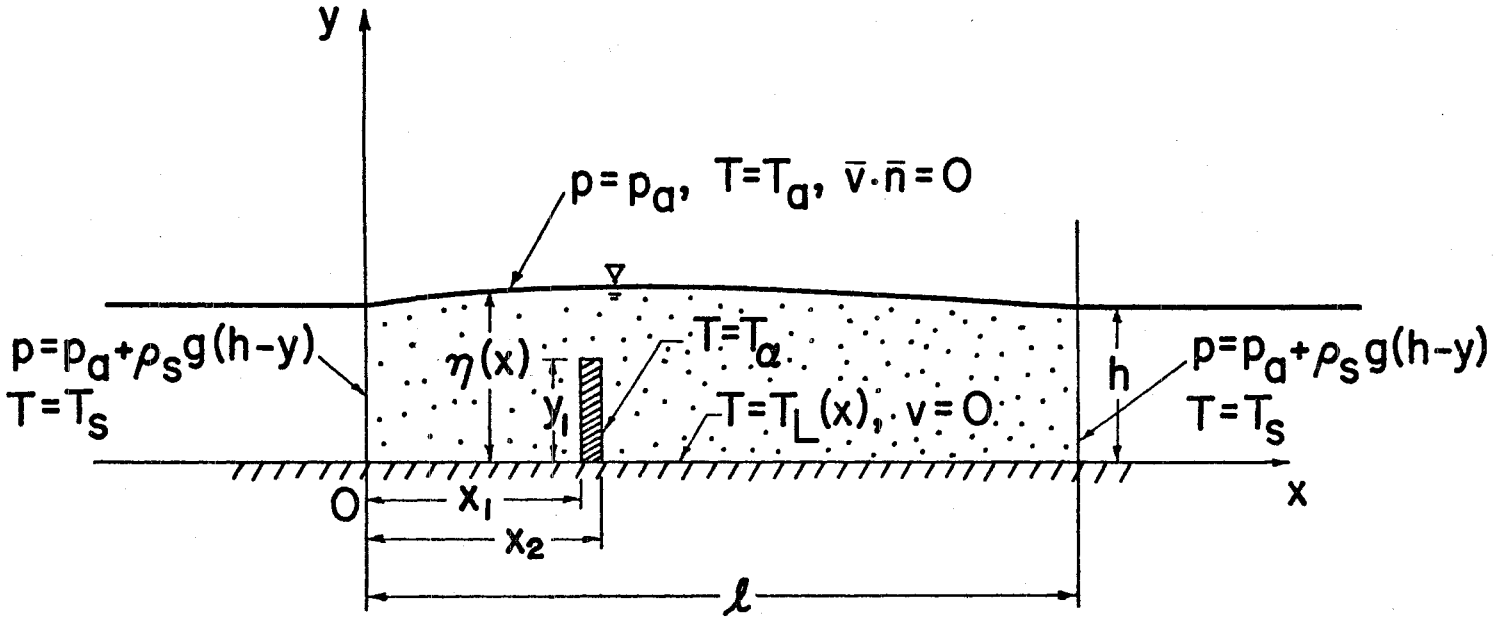

Fig. 1B IDEALIZED MODEL OF A GEOTHERMAL RESERVOIR WITH DIKE INTRUSION

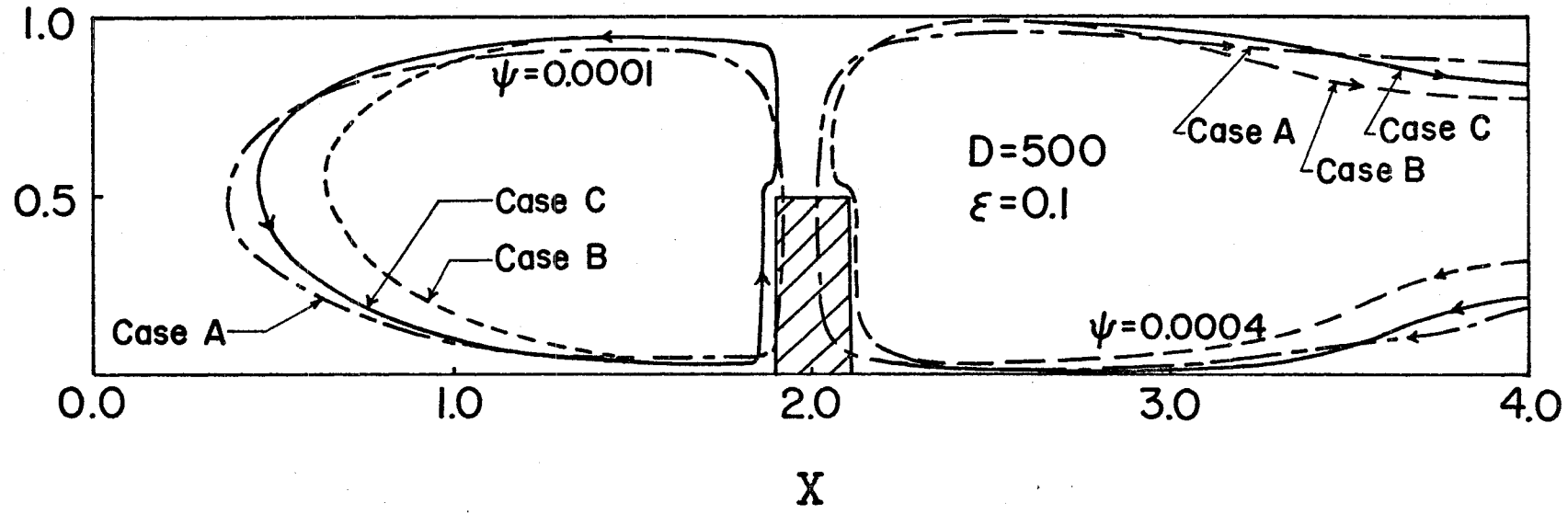

Fig. 2 EFFECTS OF VERTICAL AND HORIZONTAL HEATING ON STREAMLINES IN UNCONFINED GEOTHERMAL RESERVOIRS

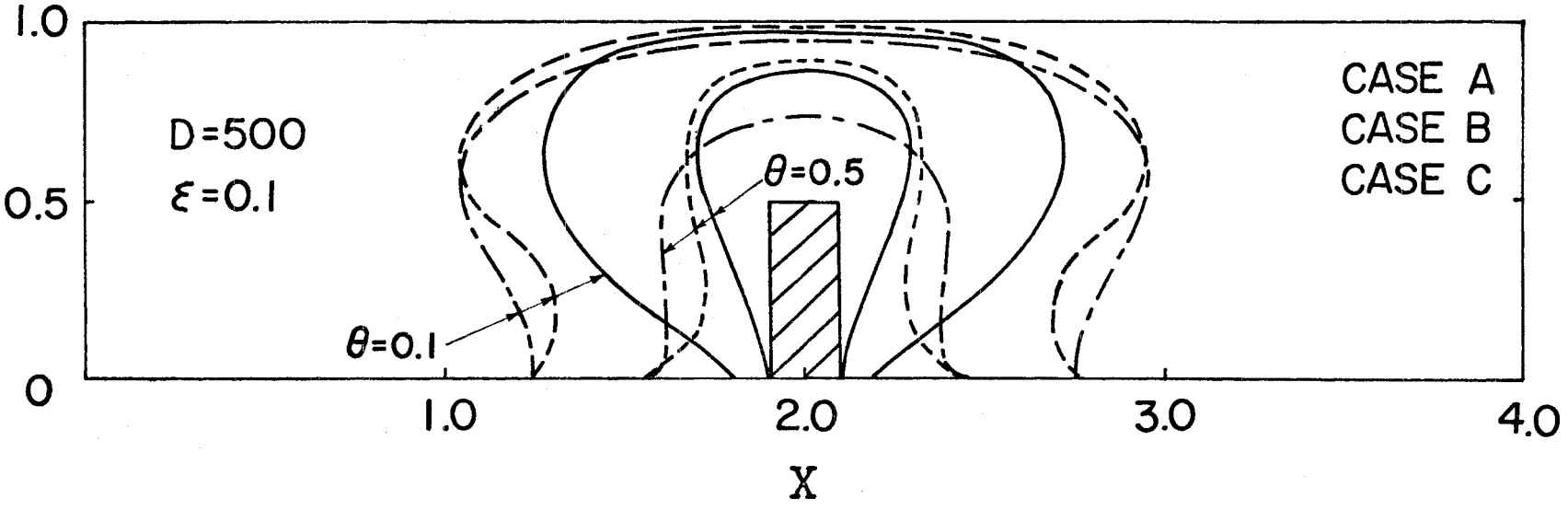

Fig. 3 EFFECTS OF VERTICAL AND HORIZONTAL HEATING ON TEMPERATURE CONTOURS IN UNCONFINED GEOTHERMAL RESERVOIRS

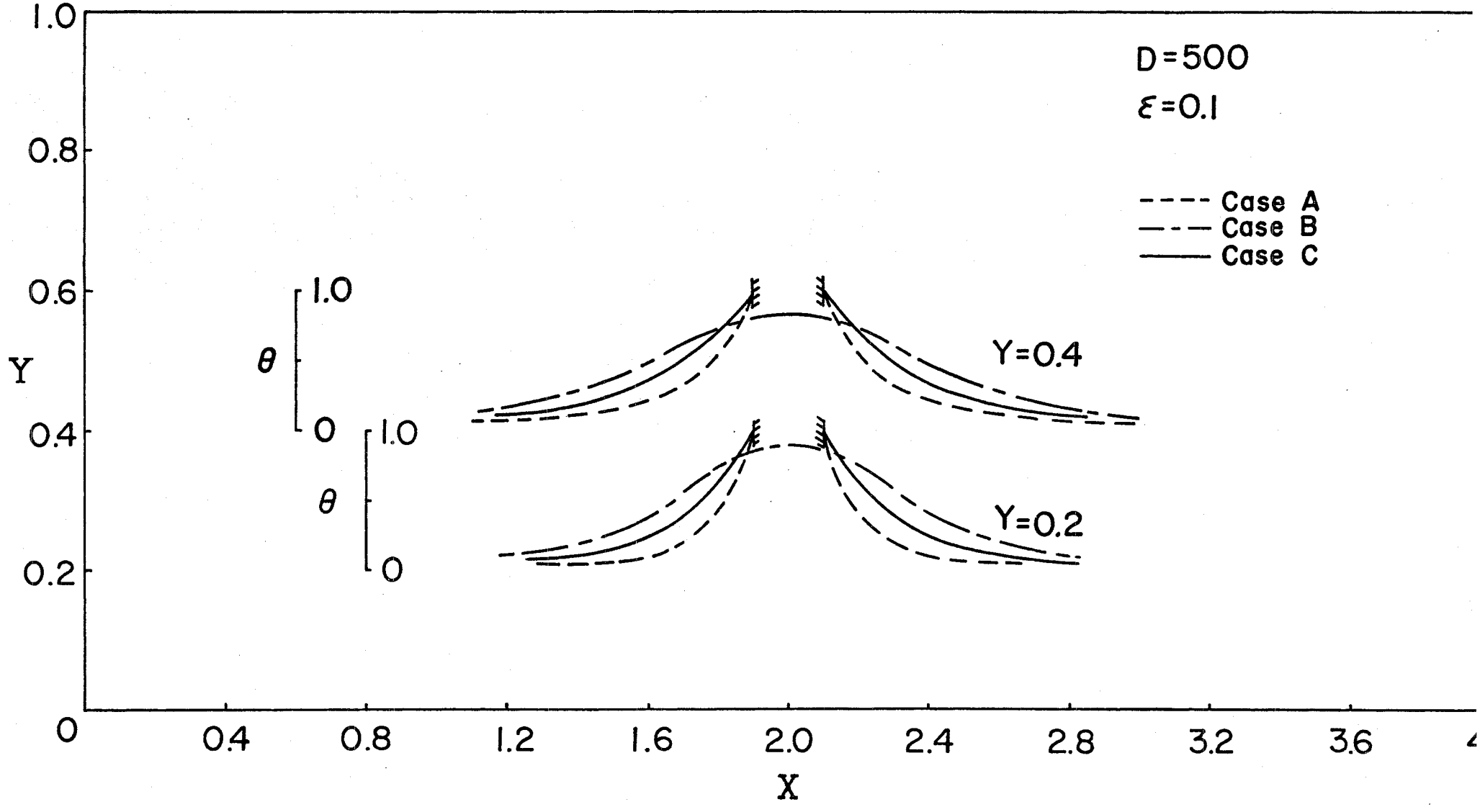

Fig. 4 HORIZONTAL TEMPERATURE DISTRIBUTION FOR CASES A, B, AND C

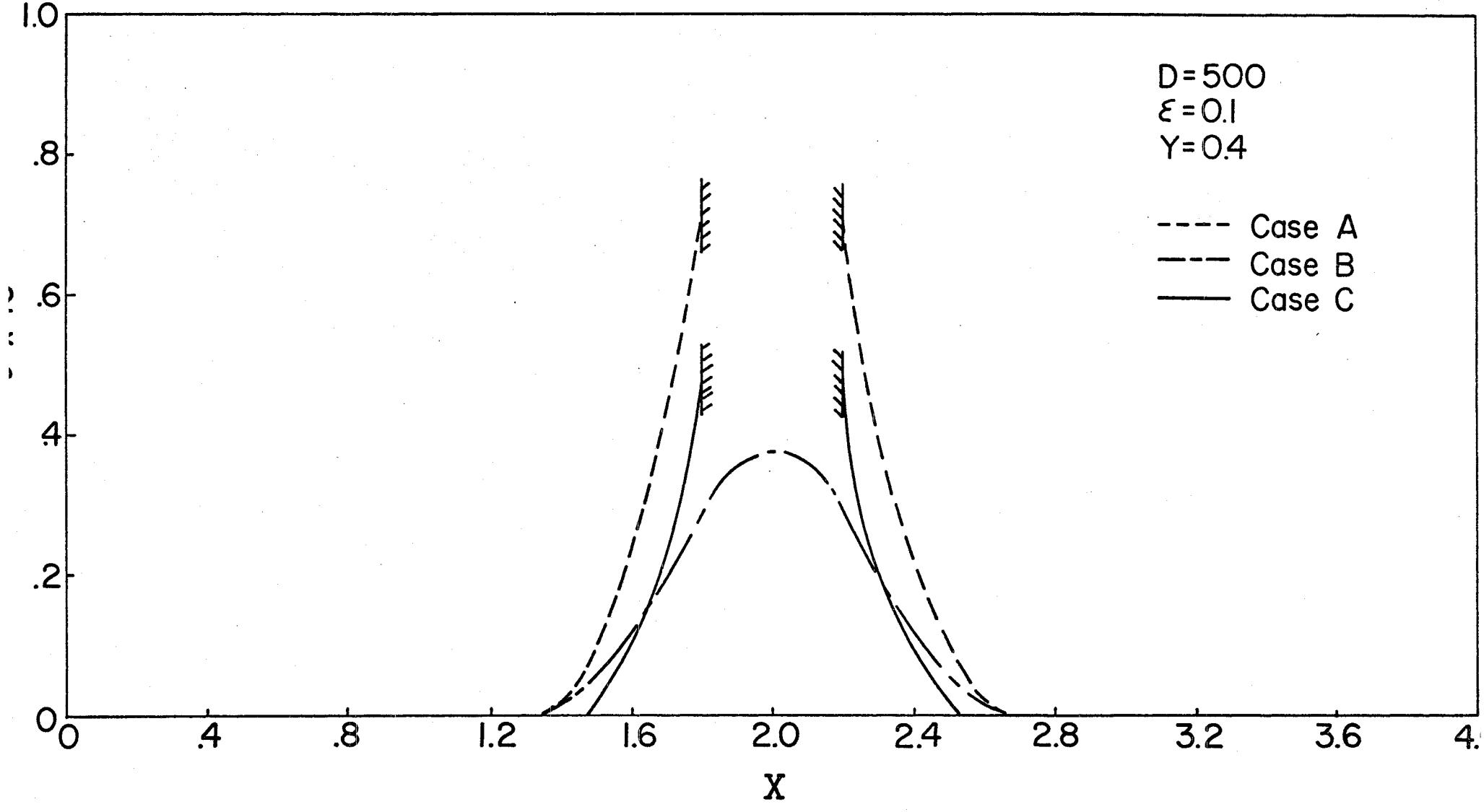

Fig. 5 VERTICAL VELOCITY PROFILES AT Y = 0.4 for CASES B, B, AND C

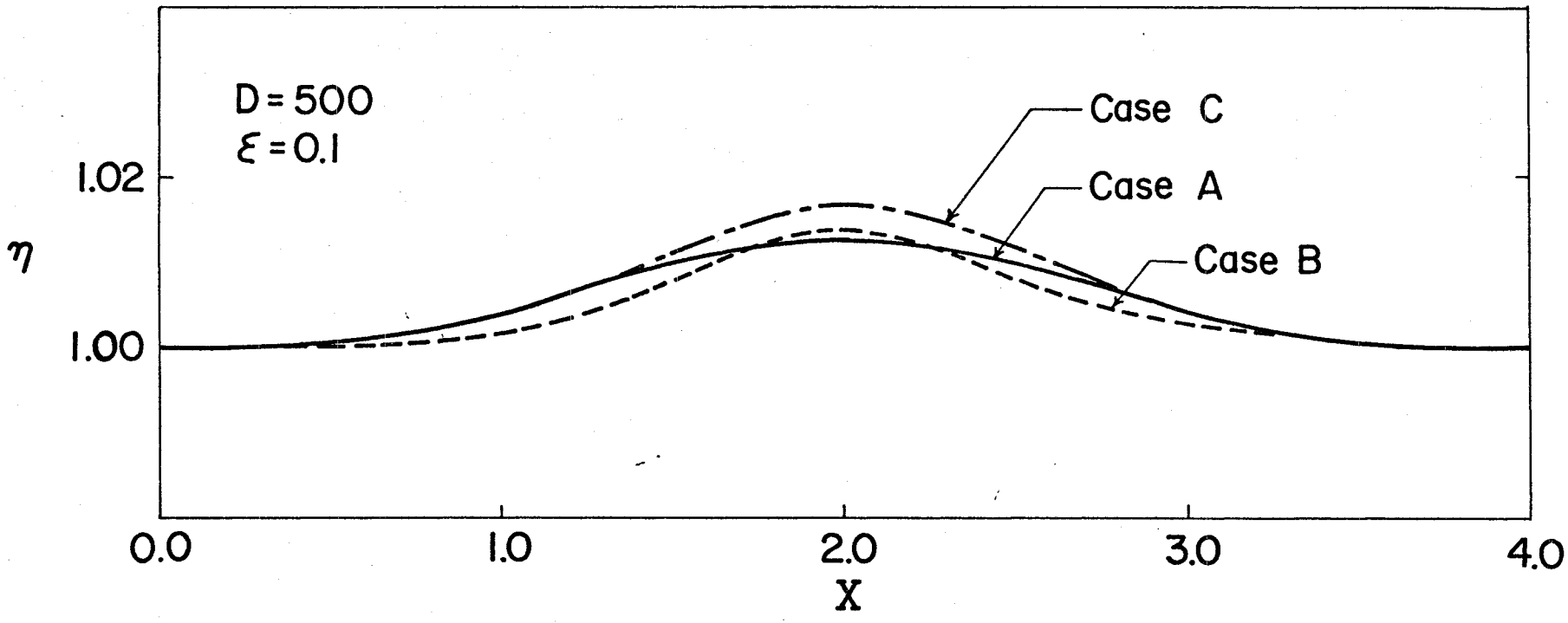

Fig. 6 EFFECTS OF HEAT SOURCES ON THE UPWELLING OF WATER TABLE FOR CASES A, B, AND C

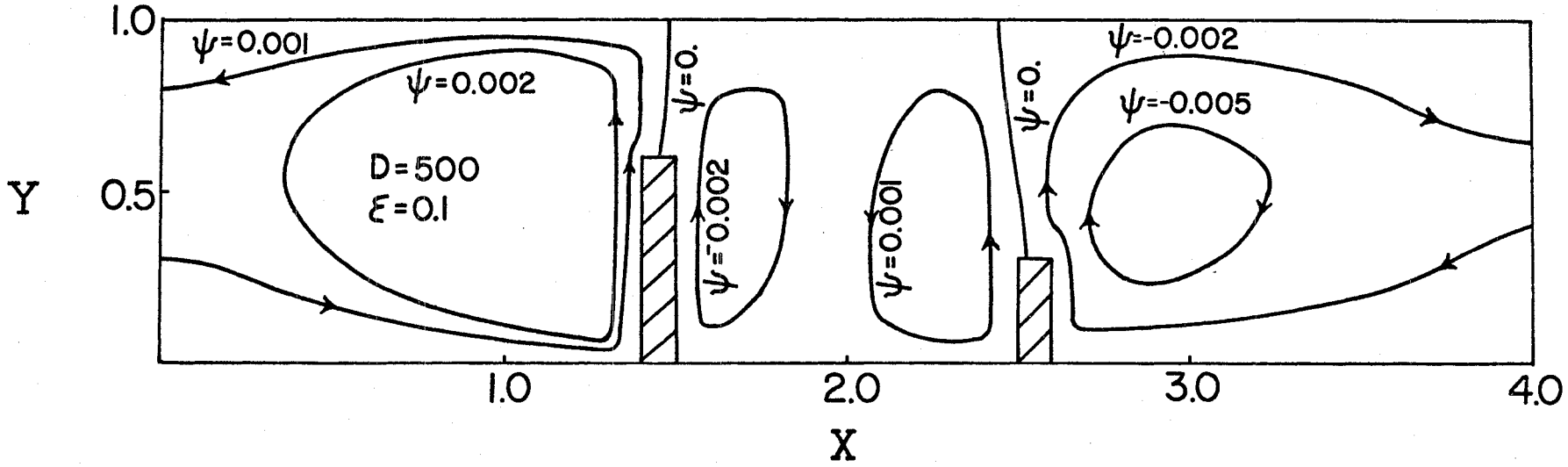

Fig. 7A CONVECTIVE PATTERN FOR CASE D

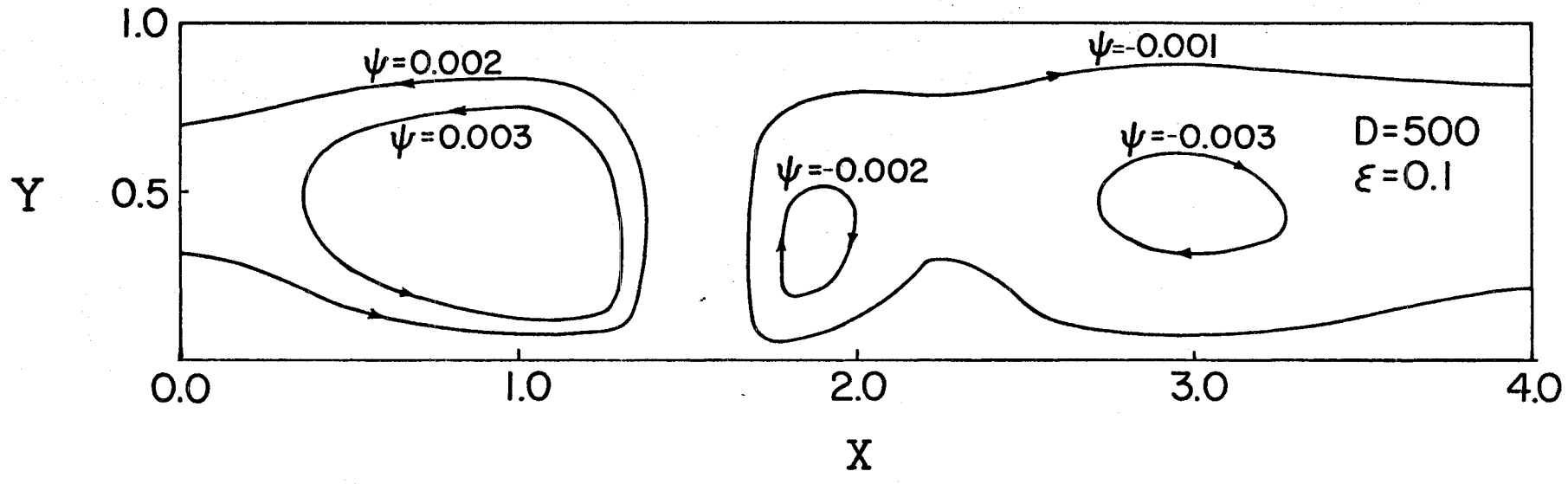

Fig. 7B CONVECTIVE PATTERN FOR CASE E

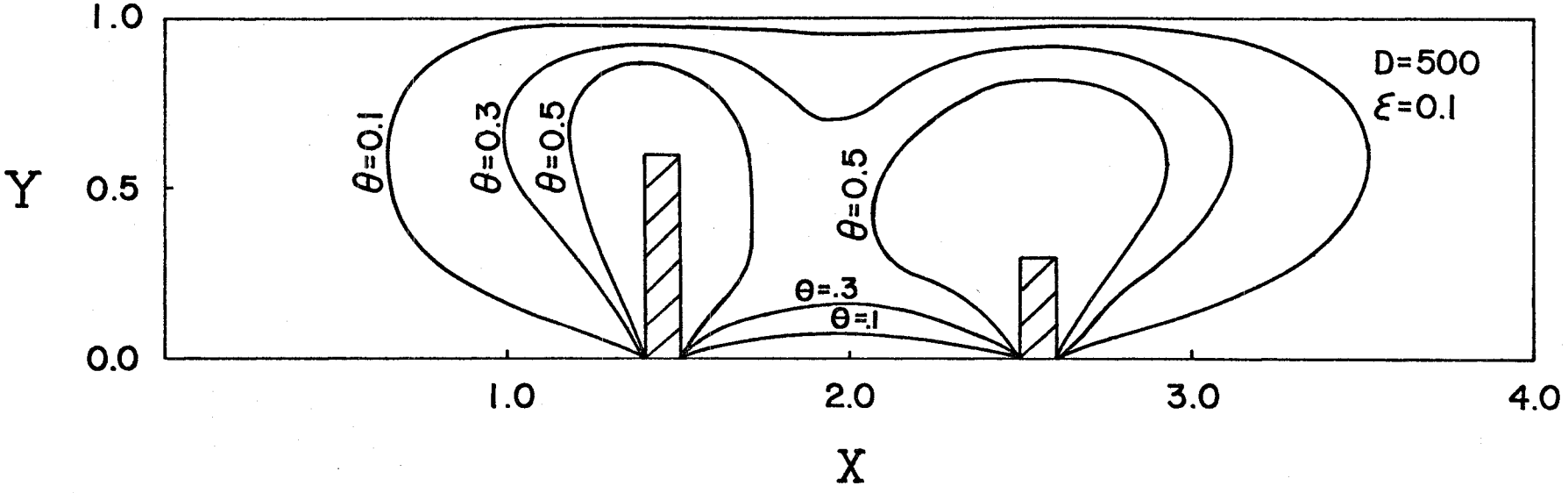

Fig. 8A TEMPERATURE CONTOURS FOR CASE D

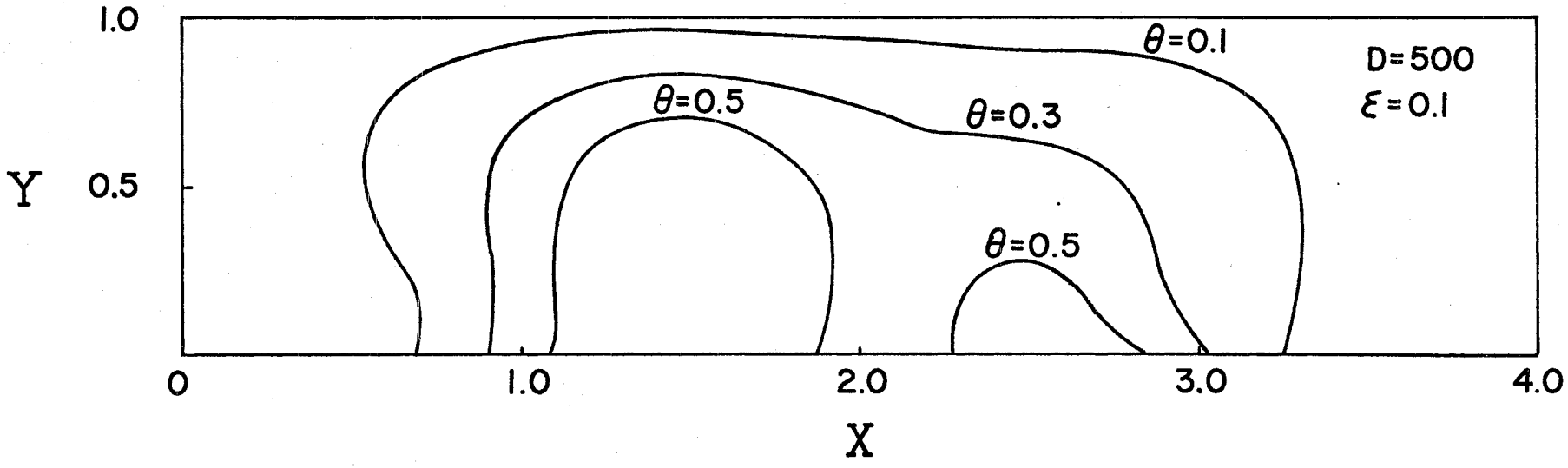

Fig. 8B TEMPERATURE CONTOURS FOR CASE E

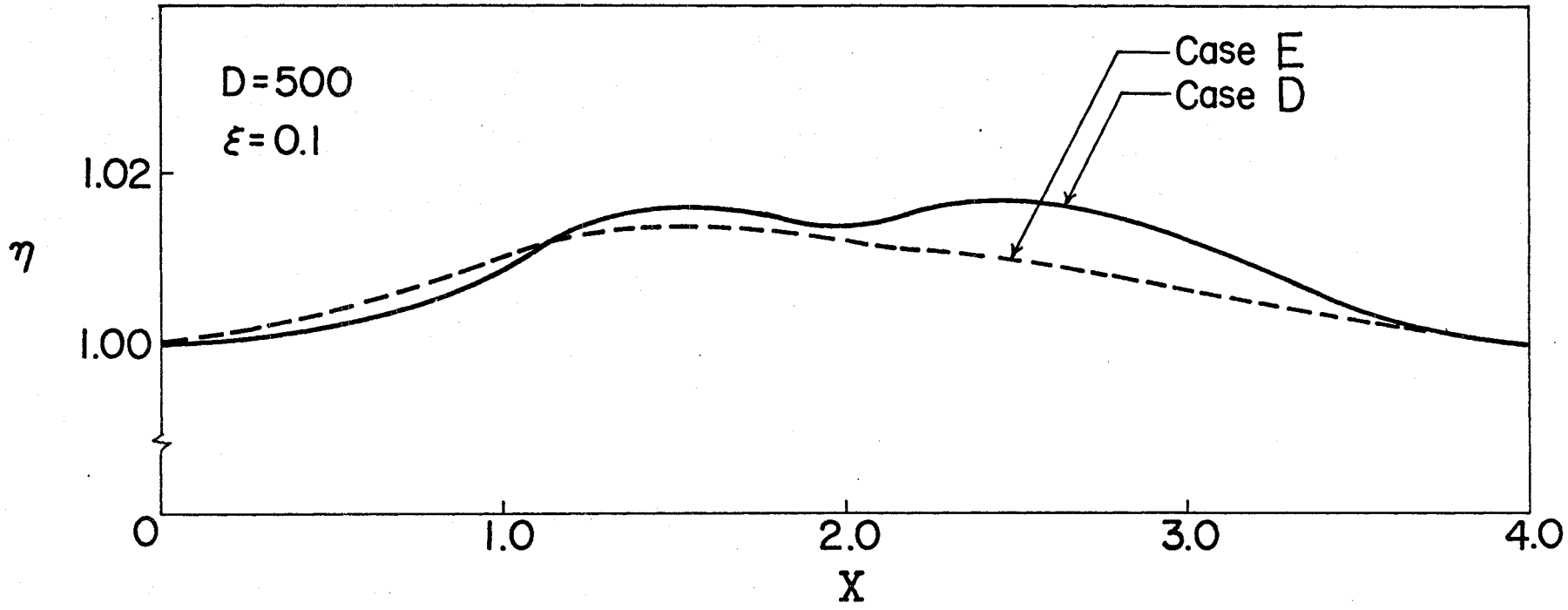

Fig. 9 EFFECTS OF HEAT SOURCES ON THE UPWELLING OF WATER TABLE FOR CASES D AND E

COMPILED BY W-2500 FORTRAN IV, REV. 6 PAGE A1

```
0001: C
0002: C      THIS PROGRAM ITERATES THE TEMPERATURE ZERO-ORDER SOLUTION
0003: C      WITH A VERTICAL SOURCE
0004: C
0005: C
0006: C      OPEN AND CREATE ARE WESTINGHOUSE FILE MANAGEMENT SYSTEM ROUTINES
0007: C      WHEN USING STANDARD FORTRAN COMPILER, FOR EXAMPLE, IBM FORTRAN G
0008: C      THIS IS TO BE DONE BY DD STATEMENT (SEE JCL LANGUAGE), HOWEVER,
0009: C      LOGICAL UNIT NUMBER SHOULD BE CHANGED TO TWO DIGITS NUMBER.
0010: C      EXAMPLE: UNIT NUMBER 100 IN WESTINGHOUSE SHOULD BE CHANGED TO 10
0011: C      WHEN USING FORTRAN G COMPILER
0012: C
0013:        DIMENSION T(41,11)
0014:        DATA TV/0.02/
0015:   2    FORMAT(2X,10F10.4)
0016: 3      FORMAT(4F5.0)
0017: 4      FORMAT('0THE VERTICAL SOURCE IS AT:','Y <',F5.1,',',F5.1,'< X <',
0018:       1 F5.1,3I5)
0019:        IIN=2
0020:        IOUT=3
0021:        M=41
0022:        N=11
0023:        M1=M-1
0024:        N1=N-1
0025:        DH=1./FLOAT(N1)
0026:        ITER=0
0027:        DERR=0.0001
0028: C
0029: C      READ IN VERTICAL HEAT SOURCE LOCATIONS
0030: C
0031:        READ(IIN,3)YS,XS1,XS2
0032:        READ(IIN,3)YS2,XS21,XS22
0033:        KYS=IFIX(YS/DH+0.1)+1
0034:        KXS1=IFIX(XS1/DH+0.1)+1
0035:        KXS2=IFIX(XS2/DH+0.1)+1
0036:        LYS=IFIX(YS2/DH+0.1)+1
0037:        LXS1=IFIX(XS21/DH+0.1)+1
0038:        LXS2=IFIX(XS22/DH+0.1)+1
0039:        WRITE(IOUT,4)YS,XS1,XS2,KYS,KXS1,KXS2
0040:        WRITE(IOUT,4)YS2,XS21,XS22,LYS,LXS1,LXS2
0041:        READ(IIN,3)XL1,XK1,XL2,XK2
0042:        READ(IIN,3)F1,F2
0043:        DO 10 I=1,M
0044:        DO 10 J=1,N
0045:        IF(I.LE.KXS2.AND.I.GE.KXS1)GO TO 11
0046:        IF(I.LE.LXS2.AND.I.GE.LXS1)GO TO 12
0047: 13     T(I,J)=0.
0048:        GO TO 10
0049: 11     IF(J.GT.KYS)GO TO 13
0050:        GO TO 14
0051: 12     IF(J.GT.LYS)GO TO 13
0052: 14     T(I,J)=1.0
0053: 10     CONTINUE
0054:        X=0.0
```

```
0055:          XMD=(XL1+XL2)/2.
0056:          DO 18 I=1,M
0057:          IF(I.LE.KXS2.AND.I.GE.KXS1)GO TO 18
0058:          IF(I.LE.LXS2.AND.I.GE.LXS1)GO TO 18
0059:          IF(X.GT.XMD)GO TO 16
0060:          XKL=((X-XL1)/XK1)**2
0061:          GO TO 17
0062: 16       XKL=((X-XL2)/XK2)**2
0063: 17       IF(XKL.GT.9.)GO TO 19
0064:          T(I,1)=1./EXP(XKL)
0065:          IF(X.GT.XMD)GO TO 24
0066:          T(I,1)=T(I,1)*F1
0067:          GO TO 18
0068: 24       T(I,1)=F2*T(I,1)
0069:          GO TO 18
0070: 19       T(I,1)=0.
0071: 18       X=X+DH
0072:          DO 30 I=2,M1
0073:  30      T(I,N)=TV
0074:          WRITE(IOUT,150)
0075:  150     FORMAT('1','INITIAL VALUES')
0076:          WRITE(IOUT,151)((T(I,J),J=1,N),I=1,M)
0077:  151     FORMAT(2X,11F11.3)
0078: C
0079: C       PERFORM ITERATION
0080: C
0081: 60       DMAX=0.0
0082:          FMAX=0.0
0083:          ITER=ITER+1
0084:          DO 50 I=2,M1
0085:          DO 50 J=2,N1
0086:          IF(I.LE.KXS2.AND.I.GE.KXS1)GO TO 52
0087:          IF(I.GT.LXS2.OR.I.LT.LXS1)GO TO 51
0088:          IF(J.GT.LYS)GO TO 51
0089:          GO TO 50
0090: 52       IF(J.GT.KYS)GO TO 51
0091:          GO TO 50
0092: 51       CONTINUE
0093:          TEMP=T(I,J)
0094:          T(I,J)=(T(I+1,J)+T(I-1,J)+T(I,J+1)+T(I,J-1))/4.
0095:          DIFF=ABS(T(I,J)-TEMP)
0096:          DABV=ABS(T(I,J))
0097:          IF(DABV.GT.FMAX)FMAX=DABV
0098:          IF(DIFF.GT.DMAX)DMAX=DIFF
0099:  50      CONTINUE
0100:          DDIF=DMAX/FMAX
0101: C
0102: C       TEST FOR ACCURACY IF REACHED GO TO NEXT, ELSE GO TO ITERATION
0103: C
0104:          IF(DDIF.GT.DERR) GO TO 60
0105:          WRITE(IOUT,152) ITER,FMAX,DMAX,DDIF
0106:  152     FORMAT('1','CONVERGENCE ON ITERATION ',I3,2X, 'FUNCTIONAL MAXIMUM
0107:        C ',F10.3/ 2X, 'ABS MAXIMUM DIFFERENCE', F6.3,2X, 'PERCENTAGE DIFFE
```

```
0108:       CRENCE ', F6.4)
0109:        WRITE(IOUT,153)XK,XL
0110:  153   FORMAT('0',  'TEMPERATURE O  WITH XK=', F4.2, ' AND XL=',F4.2//)
0111:        WRITE(IOUT,151)((T(I,J),J=1,N),I=1,M)
0112:        CALL CREATE(100,'TEMOFILE')
0113:        DO 1010 I=1,M
0114: 1010   WRITE(100)(T(I,J),J=1,N)
0115:        STOP
0116:        END
```

```
0001:       DIMENSION T(41,11),TY(41,11)
0002:       DATA M/41/,N/11/,IN/2/,IO/3/
0003: C
0004: C     THIS PROGRAM CALCULATES THE PARTIAL DERIVATIVES OF TO
0005: C     CONVECTION WITH VERTICAL HEAT SOURCE
0006: C
0007: C
0008: C
0009: C
0010:       CALL OPEN (100,'TEMOFILE')
0011:       DO 999 I=1,M
0012: 999   READ(100)(T(I,J),J=1,N)
0013:       WRITE(IO,1)
0014: 1     FORMAT('1','   THE TO VALUES'///)
0015:       IZ=N/10
0016:       WRITE(IO,2)((T(I,J),J=1,N,IZ),I=1,M,IZ)
0017: 2     FORMAT(' ',3X,11F11.3)
0018:       DH=1./FLOAT(N-1)
0019: C
0020: C     READ IN VERTICAL SOURCE LOCATIONS
0021: C
0022:       READ(IN,5)YS1,XS1,XS2
0023:       READ(IN,5)YS2,XSA,XSB
0024: 5     FORMAT(3F5.0)
0025:       KYS= IFIX(YS1/DH+0.1)+1
0026:       KXS1=IFIX(XS1/DH+0.1)+1
0027:       KXS2=IFIX(XS2/DH+0.1)+1
0028:       LYS= IFIX(YS2/DH+0.1)+1
0029:       LXS1=IFIX(XSA/DH+0.1)+1
0030:       LXS2=IFIX(XSB/DH+0.1)+1
0031: C
0032: C     FIND  D(TO)/DY
0033: C
0034:       DO 50 I=1,M
0035:       DO 50 J=1,N
0036:       IF(I.GT.KXS1.AND.I.LT.KXS2)GO TO 53
0037:       IF(I.GT.LXS1.AND.I.LT.LXS2)GO TO 54
0038: 56    IF(J.EQ.1)GO TO 51
0039:       IF(J.EQ.N)GO TO 52
0040:       TY(I,J)=0.5*(T(I,J+1)-T(I,J-1))/DH
0041:       GO TO 50
0042: 51    TY(I,J)=(T(I,J+1)-T(1,J))/DH
0043:       GO TO 50
0044: 52    TY(I,J)=(T(I,J)-T(I,J-1))/DH
0045:       GO TO 50
0046: 53    IF(J.GT.KYS)GO TO 56
0047:       IF(J.EQ.KYS)GO TO 51
0048:       TY(I,J)=0.
0049:       GO TO 50
0050: 54    IF(J.GT.LYS)GO TO 56
0051:       IF(J.EQ.LYS)GO TO 51
0052:       TY(I,J)=0.
0053: 50    CONTINUE
0054:       WRITE(IO,4)
```

```
0055: 4      FORMAT('1','   THE VALUES OF  D(TO)/DY'///)
0056:        CALL CREATE(100,'TODYFILE')
0057:        DO 60 I=1,M
0058:        WRITE(100)(TY(I,J),J=1,N)
0059: 60     CONTINUE
0060:        WRITE(IO,2)((TY(I,J),J=1,N),I=1,M)
0061:        M1=M-1
0062:        DO 70 J=1,N
0063:        TY(1,J)=0.5*(4.*T(2,J)-3.*T(1,J)-T(3,J))/DH
0064:        TY(M,J)=0.5*(3.*T(M,J)-4.*T(M1 ,J)+T(M-2,J))/DH
0065:        DO 70 I=2,M1
0066:        IF(I.GE.KXS1.AND.I.LE.KXS2)GO TO 71
0067:        IF(I.GE.LXS1.AND.I.LE.LXS2)GO TO 72
0068: 73     TY(I,J)=0.5*(T(I+1,J)-T(I-1,J))/DH
0069:        GO TO 70
0070: 71     IF(J.GE.KYS)GO TO 73
0071:        IF(I.EQ.KXS1)GO TO 74
0072:        IF(I.EQ.KXS2)GO TO 75
0073:        TY(I,J)=0.
0074:        GO TO 70
0075: 74     TY(I,J)=(T(I,J)-T(I-1,J))/DH
0076:        GO TO 70
0077: 75     TY(I,J)=(T(I+1,J)-T(I,J))/DH
0078:        GO TO 70
0079: 72     IF(J.GE.LYS)GO TO 73
0080:        IF(I.EQ.LXS2)GO TO 75
0081:        IF(I.EQ.LXS1)GO TO 74
0082:        TY(I,J)=0.
0083: 70     CONTINUE
0084:        CALL CREATE(100,'TODXFILE')
0085:        DO 80 I=1,M
0086:        WRITE(100)(TY(I,J),J=1,N)
0087: 80     CONTINUE
0088:        WRITE(IO,3)
0089: 3      FORMAT('1    THE VALUES OF D(TO)/DX'///)
0090:        WRITE(IO,2)((TY(I,J),J=1,N,IZ),I=1,M,IZ)
0091:        CALL EXIT
0092:        END
```

```
0001: C
0002: C     THIS PROGRAM ITERATES THE PRESSURE FIRST-ORDER SOLUTION
0003: C     WITH A VERTICAL SOURCE
0004: C
0005: C
0006: C     SEE TEMPERATURE ZEROTH ORDER PROGRAM FOR INFORMATION ABOUT
0007: C     WESTINGHOUSE FMS OPEN AND CREATE ROUTINES
0008: C
0009:       DIMENSION P(41,11),T(41),TY(41,11)
0010:       DATA P/451*0.0/
0011:       IN=2
0012:       IO=3
0013:       M=41
0014:       N=11
0015:       M1=M-1
0016:       N1=N-1
0017:       DH=1.0/FLOAT(N1)
0018:       DHSQ=DH*DH
0019:       ITER=0
0020:       DERR=0.001
0021:       READ(IN,2)YS,XS1,XS2
0022:       READ(IN,2)YS2,XS21,XS22
0023: 2     FORMAT(3F5.0)
0024:       KYS=IFIX(YS/DH+0.1)+1
0025:       KXS1=IFIX(XS1/DH+0.1)+1
0026:       KXS2=IFIX(XS2/DH+0.1)+1
0027:       LYS=IFIX(YS2/DH+0.1)+1
0028:       LXS1=IFIX(XS21/DH+0.1)+1
0029:       LXS2=IFIX(XS22/DH+0.1)+1
0030:       WRITE(IO,4)YS,XS1,XS2
0031:       WRITE(IO,4)YS2,XS21,XS22
0032: 4     FORMAT('0','THE VERTICAL SOURCE IS AT:  Y <',F5.1,' ,',F5.1,' < X'
0033:      1  ,' <',F5.1)
0034:       CALL OPEN(105,'TEMOFILE')
0035:       DO 10 I=1,M
0036:       READ(105)(TY(I,J),J=1,N)
0037:       T(I)=TY(I,1)
0038: 10    CONTINUE
0039:       CALL OPEN(100,'TODYFILE')
0040:       DO 12 I=1,M
0041:       READ(100) (TY(I,J),J=1,N)
0042:    12 CONTINUE
0043:     15 DMAX=0.0
0044:       FMAX=0.0
0045:       DO 20 I=2,M1
0046:       DO 20 J=1,N
0047:       TEMP=P(I,J)
0048:       IF(I.GE.KXS1.AND.I.LE.KXS2)GO TO 201
0049:       IF(I.GE.LXS1.AND.I.LE.LXS2)GO TO 202
0050:       GO TO 21
0051: 201   IF(J.GT.KYS)GO TO 21
0052:       IF(I.EQ.KXS1)GO TO 22
0053:       IF(I.EQ.KXS2)GO TO 23
0054:       GO TO 28
```

```
0055: 202    IF(J.GT.LYS)GO TO 21
0056:        IF(I.EQ.LXS1)GO TO 22
0057:        IF(I.EQ.LXS2)GO TO 23
0058:        GO TO 28
0059: 22     IF(J.EQ.KYS)GO TO 24
0060:        IF(J.EQ.1)GO TO 31
0061:        P(I,J)=2.*P(I-1,J)+P(I,J+1)+P(I,J-1)-DHSQ*TY(I,J)
0062:        GO TO 30
0063: 31     P(I,J)=2.*P(I-1,J)+2.*P(I,J+1)-2.*DH*T(I)-DHSQ*TY(I,J)
0064:        GO TO 30
0065: 24     P(I,J)=2.*P(I-1,J)+2.*P(I,J+1)-2.*DH-DHSQ *TY(I,J)
0066:        GO TO 30
0067: 23     IF(J.EQ.KYS)GO TO 27
0068:        IF(J.EQ.1)GO TO 32
0069:        P(I,J)=2.*P(I+1,J)+P(I,J+1)+P(I,J-1)-DHSQ *TY(I,J)
0070:        GO TO 30
0071: 32     P(I,J)=2.*P(I+1,J)+2.*P(I,J+1)-2.*DH*T(I)-DHSQ*TY(I,J)
0072:        GO TO 30
0073: 27     P(I,J)=2.*P(I+1,J)+2.*P(I,J+1)-2.*DH-DHSQ *TY(I,J)
0074:        GO TO 30
0075: 28     IF(J.NE.KYS)GO TO 20
0076:        P(I,J)=2.*P(I,J+1)-2.*DH+P(I-1,J)+P(I+1,J)-DHSQ *TY(I,J)
0077:        GO TO 30
0078: 21     IF(J.EQ.1)GO TO 25
0079:        IF(J.EQ.N)GO TO 26
0080:        P(I,J)=P(I+1,J)+P(I-1,J)+P(I,J-1)+P(I,J+1)-DHSQ *TY(I,J)
0081:        GO TO 30
0082: 25     P(I,1)=P(I+1,1)+P(I-1,1)+2.*P(I,2)-2.*DH*T(I)-DHSQ *TY(I,1)
0083:        GO TO 30
0084: 26     P(I,N)=P(I+1,N)+P(I-1,N)+2.*P(I,N1)+2.*DH*0.02-DHSQ *TY(I,N)
0085: 30     P(I,J)=0.25*P(I,J)
0086:        DABV=ABS(P(I,J)-TEMP)
0087:        DFAB=ABS(P(I,J))
0088:        IF(DABV.GT.DMAX)DMAX=DABV
0089:        IF(DFAB.GT.FMAX)FMAX=DFAB
0090:    20 CONTINUE
0091:        DABV=DMAX/FMAX
0092:        ITER=ITER+1
0093:        IF(DABV.GT.DERR)GO TO 15
0094:        WRITE(IO,5)
0095:   5    FORMAT('1','   THE PRESURE FUNCTION - P1'///)
0096:        DO 50 I=1,M
0097:   50   WRITE(IO,6)(P(I,J),J=1,N)
0098:   6    FORMAT(' ',3X,11F11.3)
0099:        WRITE(IO,7)ITER
0100: 7      FORMAT(//// 4X,'ITERATION CONVERGES AFTER',I4,'TRIES')
0101:        CALL CREATE(102,'PRS1FILE')
0102:        DO 100 I=1,M
0103:        WRITE(102)(P(I,J),J=1,N)
0104: 100    CONTINUE
0105:        STOP
0106:        END
```

```
0001:       DIMENSION T(41,11),TY(41,11),TA(41),TB(41)
0002:       DATA M/41/,N/11/,IN/2/,IO/3/
0003: C
0004: C     THIS PROGRAM CALCULATES THE PARTIAL DERIVATIVES OF P1
0005: C     VERTICAL HEAT SOURCE
0006: C
0007:       CALL OPEN(101,'PRS1FILE')
0008:       DO 200 I=1,M
0009:       READ(101)(T(I,J),J=1,N)
0010: 200   CONTINUE
0011:       IZ=N/10
0012:       WRITE(IO,1)
0013: 1     FORMAT('1','    THE P1 VALUES'///)
0014:       WRITE(IO,2)((T(I,J),J=1,N,IZ),I=1,M,IZ)
0015: 2     FORMAT(' ',3X,11F11.3)
0016: C     READ IN VERTICAL SOURCE LOCATIONS
0017:       DH=1./FLOAT(N-1)
0018:       READ(IN,5)YS1,XS1,XS2
0019:       READ(IN,5)YS2,XSA,XSB
0020: 5     FORMAT(3F5.0)
0021:       KYS= IFIX(YS1/DH+0.1)+1
0022:       KXS1=IFIX(XS1/DH+0.1)+1
0023:       KXS2=IFIX(XS2/DH+0.1)+1
0024:       LYS= IFIX(YS2/DH+0.1)+1
0025:       LXS1=IFIX(XSA/DH+0.1)+1
0026:       LXS2=IFIX(XSB/DH+0.1)+1
0027: C     READ IN TEMPERATURE 0 VALUES
0028:       CALL OPEN(102,'TEMOFILE')
0029:       DO 210 I=1,M
0030:       READ(102)(TY(I,J),J=1,N)
0031: 210   CONTINUE
0032:       DO 20 I=1,M
0033:       TA(I)=TY(I,N)
0034:       TB(I)=TY(I,1)
0035: 20    CONTINUE
0036: C     FIND  D(P1)/DY
0037:       DO 50 I=1,M
0038:       DO 50 J=1,N
0039:       IF(J.EQ.1)GO TO 51
0040:       IF(J.EQ.N)GO TO 52
0041:       IF(I.GT.KXS1.AND.I.LT.KXS2)GO TO 53
0042:       IF(I.GT.LXS1.AND.I.LT.LXS2)GO TO 54
0043: 56    TY(I,J)=(T(I,J+1)-T(I,J-1))/DH*0.5
0044:       GO TO 50
0045: 51    TY(I,J)=TB(I)
0046:       GO TO 50
0047: 52    TY(I,J)=TA(I)
0048:       GO TO 50
0049: 53    IF(J.GT.KYS)GO TO 56
0050:       IF(J.EQ.KYS)GO TO 57
0051:       TY(I,J)=0.
0052:       GO TO 50
0053: 57    TY(I,J)=1.
0054:       GO TO 50
```

```
0055: 54     IF(J.GT.LYS)GO TO 56
0056:        IF(J.EQ.LYS)GO TO 58
0057:        TY(I,J)=0.
0058:        GO TO 50
0059: 58     TY(I,J)=1.
0060: 50     CONTINUE
0061:        WRITE(IO,4)
0062: 4      FORMAT('1','THE VALUES OF D(P1)/DY'///)
0063:        CALL CREATE(100,'P1DYFILE')
0064:        DO 60 I=1,M
0065:        WRITE(100)(TY(I,J),J=1,N)
0066: 60     CONTINUE
0067:        WRITE(IO,2)((TY(I,J),J=1,N,IZ),I=1,M,IZ)
0068:        M1=M-1
0069:        DO 70 J=1,N
0070:        TY(1,J)=0.5*(4.*T(2,J)-3.*T(1,J)-T(3,J))/DH
0071:        TY(M,J)=0.5*(3.*T(M,J)-4.*T(M1 ,J)+T(M-2,J))/DH
0072:        DO 70 I=2,M1
0073:        IF(I.GE.KXS1.AND.I.LE.KXS2)GO TO 71
0074:        IF(I.GE.LXS1.AND.I.LE.LXS2)GO TO 72
0075: 73     TY(I,J)=0.5*(T(I+1,J)-T(I-1,J))/DH
0076:        GO TO 70
0077: 71     IF(J.GE.KYS)GO TO 73
0078:        TY(I,J)=0.
0079:        GO TO 70
0080: 72     IF(J.GE.LYS)GO TO 73
0081:        TY(I,J)=0.
0082: 70     CONTINUE
0083:        CALL CREATE(103,'P1DXFILE')
0084:        DO 80 I=1,M
0085:        WRITE(103)(TY(I,J),J=1,N)
0086: 80     CONTINUE
0087:        WRITE(IO,3)
0088: 3      FORMAT('1    THE VALUES OF D(P1)/DX'///)
0089:        WRITE(IO,2)((TY(I,J),J=1,N,IZ),I=1,M,IZ)
0090:        CALL EXIT
0091:        END
```

```
0001: C
0002: C     THIS PROGRAM CALCULATES THE ADVECTION TERM IN THE THETA 1 EQUATION
0003: C     SEE TEMPERATURE ZEROTH ORDER PROGRAM FOR INFORMATION ABOUT
0004: C     WESTINGHOUSE FMS OPEN AND CREATE ROUTINES
0005: C
0006:       DIMENSION CON(41,11),FA(41,11),FB(41,11)
0007:       DATA M/41/,N/11/,IN/2/,IO/3/
0008: 1     FORMAT(' ',3X,11F11.3)
0009:       ITER=0
0010:       DISCH=500.
0011:       M1=M-1
0012:       N1=N-1
0013:       DH=1./FLOAT(N1)
0014:       CALL OPEN(100,'P1DXFILE')
0015:       READ(100)((FA(I,J),J=1,N),I=1,M)
0016:       CALL OPEN(100,'T0DXFILE')
0017:       READ(100)((FB(I,J),J=1,N),I=1,M)
0018:       DO 500 I=1,M
0019:       DO 500 J=1,N
0020:       CON(I,J)=FA(I,J)*FB(I,J)
0021: 500   CONTINUE
0022:       CALL OPEN (100,'P1DYFILE')
0023:       READ(100)((FA(I,J),J=1,N),I=1,M)
0024:       CALL OPEN(100,'T0DYFILE')
0025:       READ(100)((FB(I,J),J=1,N),I=1,M)
0026:       DO 600 I=1,M
0027:       DO 600 J=1,N
0028:       CON(I,J)=CON(I,J)+FA(I,J)*FB(I,J)
0029: 600   CONTINUE
0030:       CALL OPEN(100,'TEM0FILE')
0031:       READ(100)((FA(I,J),J=1,N),I=1,M)
0032:       DO 650 I=1,M
0033:       DO 650 J=1,N
0034:       CON(I,J)=CON(I,J)-FA(I,J)*FB(I,J)
0035:       CON(I,J)=CON(I,J)*DISCH*DH*DH
0036: 650   CONTINUE
0037:       WRITE(IO,13)
0038:  13   FORMAT('1','   THE CONSTANT FUNCTION')
0039:       WRITE(3,502) DISCH
0040: 502   FORMAT('0', 'DISCHARGE = ',F6.0)
0041:       CALL CREATE(100,'TEM1TERM')
0042:       WRITE(IO,1)((CON(I,J),J=1,N),I=1,M)
0043:       DO 610 I=1,M
0044:       WRITE(100)(CON(I,J),J=1,N)
0045: 610   CONTINUE
0046:       STOP
0047:       END
```

```
0001: C
0002: C      THIS PROGRAM ITERATES THE TEMPERATURE FIRST-ORDER SOLUTION
0003: C      WITH VERTICAL HEAT SOURCES
0004: C      SEE TEMPERATURE ZEROTH ORDER PROGRAM FOR INFORMATION ABOUT
0005: C      WESTINGHOUSE FMS OPEN AND CREATE ROUTINES
0006: C
0007:        DIMENSION FA(41,11),FB(41,11),CON(41,11)
0008:        DATA FB/451*0.0/
0009: 1      FORMAT(4X,11F11.3)
0010: 3      FORMAT(3F5.0)
0011:        DERR=0.0001
0012:        IIN=2
0013:        IO=3
0014:        M=41
0015:        N=11
0016:        M1=M-1
0017:        N1=N-1
0018:        DH=1./FLOAT(N1)
0019:        READ(IIN,3)YS,XS1,XS2
0020:        READ(IIN,3)YS2,XS21,XS22
0021:        LYS=IFIX(YS2/DH+0.1)+1
0022:        LXS1=IFIX(XS21/DH+0.1)+1
0023:        LXS2=IFIX(XS22/DH+0.1)+1
0024:        KYS=IFIX(YS/DH+0.1)+1
0025:        KXS1=IFIX(XS1/DH+0.1)+1
0026:        KXS2=IFIX(XS2/DH+0.1)+1
0027:        WRITE(IOUT,4)YS,XS1,XS2
0028:        WRITE(IOUT,4)YS2,XS21,XS22
0029: 4      FORMAT('1THE VERTICAL SOURCE IS AT:','Y <',F5.1,',',F5.1,'<X<',
0030:       1  F5.1)
0031:        CALL OPEN(103,'TODYFILE')
0032:        CALL OPEN(104,'PRS1FILE')
0033:        DO 20 I=1,M
0034:        READ(103) (FA(I,J),J=1,N)
0035:        READ(104) (CON(I,J),J=1,N)
0036:   20   FB(I,N)= -(FA(I,N) * CON(I,N))
0037:        CALL OPEN(100,'TEM1TERM')
0038:        DO 13 I=1,M
0039: 13     READ(100)(CON(I,J),J=1,N)
0040: 900    DMAX=0.
0041:        FMAX=0.
0042:        DO 700 I=2,M1
0043:        DO 700 J=2,N1
0044:        IF(I.GE.KXS1.AND.I.LE.KXS2.AND.J.LE.KYS)GO TO 18
0045:        IF(I.GE.LXS1.AND.I.LE.LXS2.AND.J.LE.LYS)GO TO 18
0046:        GO TO 19
0047: 18     FB(I,J)=0.0
0048:         GO TO 700
0049: 19     TEMP=FB(I,J)
0050:        FB(I,J)=FB(I+1,J)+FB(I-1,J)+FB(I,J+1)+FB(I,J-1)+CON(I,J)
0051:        FB(I,J)=0.25*FB(I,J)
0052:        DAB=ABS(FB(I,J)-TEMP)
0053:        FAB=ABS(FB(I,J))
0054:        IF(DAB.GT.DMAX)DMAX=DAB
```

```
055:          IF(FAB.GT.FMAX)FMAX=FAB
056: 700      CONTINUE
057:          ITER=ITER+1
058:          DAB=DMAX/FMAX
059:          IF(DAB.GT.DERR)GO TO 900
060: 11       FORMAT('1     THE T1 FUNCTION'///)
061: 12       FORMAT('0','      AFTER',I5,'  ITERATIONS')
062:          WRITE(IO,11)
063:          WRITE(IO,12)ITER
064:          WRITE(IO,1)((FB(I,J),J=1,N),I=1,M)
065:          CALL CREATE(100,'TEM1FILE')
066:          DO 910 I=1,M
067:          WRITE(100)(FB(I,J),J=1,N)
068: 910      CONTINUE
069:          STOP
070:          END
```

```
0001:       DIMENSION T(41,11),TY(41,11),TA(41),TB(41)
0002:       DATA M/41/,N/11/,IN/2/,IO/3/
0003: C
0004: C     THIS PROGRAM CALCULATES THE PARTIAL DERIVATIVES OF T1
0005: C     VERTICAL HEAT SOURCE
0006: C
0007:       CALL OPEN (100,'TEM1FILE')
0008:       DO 200 I=1,M
0009:       READ(100)(T(I,J),J=1,N)
0010: 200   CONTINUE
0011:       IZ=N/10
0012:       WRITE(IO,1)
0013: 1     FORMAT('1','   THE T1 VALUES'///)
0014:       WRITE(IO,2)((T(I,J),J=1,N,IZ),I=1,M,IZ)
0015: 2     FORMAT(' ',3X,11F11.3)
0016: C     READ IN VERTICAL SOURCE LOCATIONS
0017:       DH=1./FLOAT(N-1)
0018:       READ(IN,5)YS1,XS1,XS2
0019:       READ(IN,5)YS2,XSA,XSB
0020: 5     FORMAT(3F5.0)
0021:       KYS= IFIX(YS1/DH+0.1)+1
0022:       KXS1=IFIX(XS1/DH+0.1)+1
0023:       KXS2=IFIX(XS2/DH+0.1)+1
0024:       LYS= IFIX(YS2/DH+0.1)+1
0025:       LXS1=IFIX(XSA/DH+0.1)+1
0026:       LXS2=IFIX(XSB/DH+0.1)+1
0027: C
0028: C     FIND D(T1)/DY
0029: C
0030:       DO 50 I=1,M
0031:       DO 50 J=1,N
0032:       IF(I.GT.KXS1.AND.I.LT.KXS2)GO TO 53
0033:       IF(I.GT.LXS1.AND.I.LT.LXS2)GO TO 54
0034: 56    IF(J.EQ.1)GO TO 51
0035:       IF(J.EQ.N)GO TO 52
0036:       TY(I,J)=(T(I,J+1)-T(I,J-1))/DH*0.5
0037:       GO TO 50
0038: 51    TY(I,J)=(T(I,J+1)-T(I,J))/DH
0039:       GO TO 50
0040: 52    TY(I,J)=(T(I,J)-T(I,J-1))/DH
0041:       GO TO 50
0042: 53    IF(J.GT.KYS)GO TO 56
0043:       IF(J.EQ.KYS)GO TO 51
0044:       TY(I,J)=0.
0045:       GO TO 50
0046: 54    IF(J.GT.LYS)GO TO 56
0047:       IF(J.EQ.LYS)GO TO 51
0048:       TY(I,J)=0.
0049: 50    CONTINUE
0050:       WRITE(IO,4)
0051: 4     FORMAT('1','THE VALUES OF D(T1)/DY'///)
0052:       CALL CREATE(100,'T1DYFILE')
0053:       DO 60 I=1,M
0054:       WRITE(100)(TY(I,J),J=1,N)
```

```
0055: 60     CONTINUE
0056:        WRITE(10,2)((TY(I,J),J=1,N,IZ),I=1,M,IZ)
0057:        M1=M-1
0058:        DO 70 J=1,N
0059:        TY(1,J)=0.5*(4.*T(2,J)-3.*T(1,J)-T(3,J))/DH
0060:        TY(M,J)=0.5*(3.*T(M,J)-4.*T(M1 ,J)+T(M-2,J))/DH
0061:        DO 70 I=2,M1
0062:        IF(I.GT.KXS1.AND.I.LT.KXS2)GO TO 71
0063:        IF(I.GT.LXS1.AND.I.LT.LXS2)GO TO 72
0064:        IF(I.EQ.KXS1.OR.I.EQ.LXS1)GO TO 76
0065:        IF(I.EQ.KXS2.OR.I.EQ.LXS2)GO TO 77
0066: 73     TY(I,J)=0.5*(T(I+1,J)-T(I-1,J))/DH
0067:        GO TO 70
0068: 71     IF(J.GE.KYS)GO TO 73
0069:        TY(I,J)=0.
0070:        GO TO 70
0071: 76     TY(I,J)=(T(I,J)-T(I-1,J))/DH
0072:        GO TO 70
0073: 77     TY(I,J)=(T(I+1,J)-T(I,J))/DH
0074:        GO TO 70
0075: 72     IF(J.GE.LYS)GO TO 73
0076:        TY(I,J)=0.
0077: 70     CONTINUE
0078:        CALL CREATE(100,'T1DXFILE')
0079:        DO 80 I=1,M
0080:        WRITE(100)(TY(I,J),J=1,N)
0081: 80     CONTINUE
0082:        WRITE(10,3)
0083: 3      FORMAT('1    THE VALUES OF D(T1)/DX'///)
0084:        WRITE(10,2)((TY(I,J),J=1,N,IZ),I=1,M,IZ)
0085:        CALL EXIT
0086:        END
```

```
0001: C
0002: C      THIS PROGRAM ITERATES THE PRESSURE SECOND ORDER SOLUTION
0003: C      WITH A VERTICAL SOURCE
0004: C
0005: C
0006: C      SEE TEMPERATURE ZEROTH ORDER PROGRAM FOR INFORMATION ABOUT
0007: C      WESTINGHOUSE FMS OPEN AND CREATE ROUTINES
0008: C
0009:        DIMENSION P(41,11),T(41),TY(41,11)
0010:        DATA P/451*0.0/
0011:        IN=2
0012:        IO=3
0013:        M=41
0014:        N=11
0015:        M1=M-1
0016:        N1=N-1
0017:        DH=1.0/FLOAT(N1)
0018:        DHSQ=DH*DH
0019:        ITER=0
0020:        DERR=0.0001
0021:        READ(IN,2)YS,XS1,XS2
0022:        READ(IN,2)YS2,XS21,XS22
0023: 2      FORMAT(3F5.0)
0024:        KYS=IFIX(YS/DH+0.1)+1
0025:        KXS1=IFIX(XS1/DH+0.1)+1
0026:        KXS2=IFIX(XS2/DH+0.1)+1
0027:        LYS=IFIX(YS2/DH+0.1)+1
0028:        LXS1=IFIX(XS21/DH+0.1)+1
0029:        LXS2=IFIX(XS22/DH+0.1)+1
0030:        WRITE(IO,4)YS,XS1,XS2
0031:        WRITE(IO,4)YS2,XS21,XS22
0032: 4      FORMAT('0','THE VERTICAL SOURCE IS AT:  Y <',F5.1,' ,',F5.1,' <
0033:       1  ,' <',F5.1)
0034: C
0035: C      COMPUTE  D(P2)/DY AT (X,1) = (P1)X * (P1)X - P1 * (P1)YY
0036: C
0037:        CALL OPEN(105,'PRS1FILE')
0038:        DO 10 I=1,M
0039:        READ(105)(TY(I,J),J=1,N)
0040:        T(I)=TY(I,N)
0041: 10     CONTINUE
0042:        CALL OPEN (100,'P1DYFILE')
0043:        DO 40 I=1,M
0044:        READ(100)(TY(I,J),J=1,N)
0045:        T(I)=-T(I)*(TY(I,N)-TY(I,N-1))/DH
0046: 40     CONTINUE
0047:        CALL OPEN(100,'P1DXFILE')
0048:        DO 41 I=1,M
0049:        READ(100)(TY(I,J),J=1,N)
0050:        T(I)=T(I)+TY(I,N)*TY(I,N)
0051: 41     CONTINUE
0052:        CALL OPEN(100,'T1DYFILE')
0053:        DO 12 I=1,M
0054:        READ(100) (TY(I,J),J=1,N)
```

```
0055:    12  CONTINUE
0056:        DO 60 I=1,M
0057:        DO 60 J=1,N
0058:        TY(I,J)=DHSQ*TY(I,J)
0059: 60     CONTINUE
0060: C
0061: C      BEGIN ITERATION
0062: C
0063:    15 DMAX=0.0
0064:       FMAX=0.0
0065:        DO 20 I=2,M1
0066:        DO 20 J=1,N
0067:        TEMP=P(I,J)
0068:        IF(I.GE.KXS1.AND.I.LE.KXS2)GO TO 201
0069:        IF(I.GE.LXS1.AND.I.LE.LXS2)GO TO 202
0070:        GO TO 21
0071: 201    IF(J.GT.KYS)GO TO 21
0072:        IF(I.EQ.KXS1)GO TO 22
0073:        IF(I.EQ.KXS2)GO TO 23
0074:        GO TO 28
0075: 202    IF(J.GT.LYS)GO TO 21
0076:        IF(I.EQ.LXS1)GO TO 22
0077:        IF(I.EQ.LXS2)GO TO 23
0078:        GO TO 28
0079: 22     IF(J.EQ.KYS)GO TO 24
0080:        IF(J.EQ.1)GO TO 31
0081:        P(I,J)=2.*P(I-1,J)+P(I,J+1)+P(I,J-1)-TY(I,J)
0082:        GO TO 30
0083: 31     P(I,J)=2.*P(I-1,J)+2.*P(I,J+1)-TY(I,J)
0084:        GO TO 30
0085: 24     P(I,J)=2.*P(I-1,J)+2.*P(I,J+1)-TY(I,J)
0086:        GO TO 30
0087: 23     IF(J.EQ.KYS)GO TO 27
0088:        IF(J.EQ.1)GO TO 32
0089:        P(I,J)=2.*P(I+1,J)+P(I,J+1)+P(I,J-1)-TY(I,J)
0090:        GO TO 30
0091: 32     P(I,J)=2.*P(I+1,J)+2.*P(I,J+1)-TY(I,J)
0092:        GO TO 30
0093: 27     P(I,J)=2.*P(I+1,J)+2.*P(I,J+1)-TY(I,J)
0094:        GO TO 30
0095: 28     IF(J.NE.KYS)GO TO 20
0096:        P(I,J)=2.*P(I,J+1)+P(I-1,J)+P(I+1,J)-TY(I,J)
0097:        GO TO 30
0098: 21     IF(J.EQ.1)GO TO 25
0099:        IF(J.EQ.N)GO TO 26
0100:        P(I,J)=P(I+1,J)+P(I-1,J)+P(I,J-1)+P(I,J+1)-TY(I,J)
0101:        GO TO 30
0102: 25     P(I,1)=P(I+1,1)+P(I-1,1)+2.*P(I,2)-TY(I,J)
0103:        GO TO 30
0104: 26     P(I,N)=P(I+1,N)+P(I-1,N)+2.*P(I,N1)+2.*DH*T(I)-TY(I,J)
0105: 30     P(I,J)=0.25*P(I,J)
0106:        DABV=ABS(P(I,J)-TEMP)
0107:        DFAB=ABS(P(I,J))
```

```
0108:          IF(DABV.GT.DMAX)DMAX=DABV
0109:          IF(DFAB.GT.FMAX)FMAX=DFAB
0110:       20 CONTINUE
0111:          DABV=DMAX/FMAX
0112:          ITER=ITER+1
0113:          IF(DABV.GT.DERR)GO TO 15
0114:          WRITE(IO,5)
0115:       5  FORMAT('1','   THE PRESURE FUNCTION - P2'///)
0116:          DO 50 I=1,M
0117:       50 WRITE(IO,6)(P(I,J),J=1,N)
0118:       6  FORMAT(' ',3X,11F11.3)
0119:          WRITE(IO,7)ITER
0120: 7        FORMAT(//// 4X,'ITERATION CONVERGES AFTER',I4,'TRIES')
0121:          CALL CREATE(102,'PRS2FILE')
0122:          DO 100 I=1,M
0123:          WRITE(102)(P(I,J),J=1,N)
0124: 100      CONTINUE
0125:          STOP
0126:          END
```

```
0001:       DIMENSION T(41,11),TY(41,11),TA(41),P1(41)
0002:       DATA M/41/,N/11/,IN/2/,IO/3/
0003: C
0004: C     THIS PROGRAM CALCULATES THE PARTIAL DERIVATIVES OF P2
0005: C     VERTICAL HEAT SOURCE
0006: C
0007:       CALL OPEN(100,'PRS2FILE')
0008:       DO 200 I=1,M
0009:       READ(100)(T(I,J),J=1,N)
0010: 200   CONTINUE
0011:       IZ=N/10
0012:       WRITE(IO,1)
0013: 1     FORMAT('1','   THE P2 VALUES'///)
0014:       WRITE(IO,2)((T(I,J),J=1,N,IZ),I=1,M,IZ)
0015: 2     FORMAT(' ',3X,11F11.3)
0016: C     READ IN VERTICAL SOURCE LOCATIONS
0017:       DH=1./FLOAT(N-1)
0018:       READ(IN,5)YS1,XS1,XS2
0019:       READ(IN,5)YS2,XSA,XSB
0020: 5     FORMAT(3F5.0)
0021:       KYS= IFIX(YS1/DH+0.1)+1
0022:       KXS1=IFIX(XS1/DH+0.1)+1
0023:       KXS2=IFIX(XS2/DH+0.1)+1
0024:       LYS= IFIX(YS2/DH+0.1)+1
0025:       LXS1=IFIX(XSA/DH+0.1)+1
0026:       LXS2=IFIX(XSB/DH+0.1)+1
0027: C
0028: C     COMPUTE  D(P2)/DY  VALUES AT (X,1)
0029: C     READ  D(P1)/DY &  D(P1)/DX VALUES
0030: C
0031:       CALL OPEN(101,'P1DXFILE')
0032: C     READ P1 VALUES & D(P1)/DY VALUES
0033: C     USED TO COMPUTE THE DERIVATIVE VALUES AT (X,1)
0034:       DO 210 I=1,M
0035:       READ(101)(TY(I,J),J=1,N)
0036: 210   CONTINUE
0037:       DO 20 I=1,M
0038:       TA(I)=TY(I,N)*TY(I,N)
0039: 20    CONTINUE
0040:       CALL OPEN(102,'PRS1FILE')
0041:       DO 21 I=1,M
0042:       READ(102)(TY(I,J),J=1,N)
0043:       P1(I)=TY(I,N)
0044: 21    CONTINUE
0045:       CALL OPEN(103,'P1DYFILE')
0046:       DO 22 I=1,M
0047:       READ(103)(TY(I,J),J=1,N)
0048: 22    CONTINUE
0049:       DO 23 I=1,M
0050:       P1YY=(TY(I,N)-TY(I,N-1))/DH
0051:       TA(I)=TA(I)-P1(I)*P1YY
0052: 23    CONTINUE
0053:       DO 50 I=1,M
0054:       DO 50 J=1,N
```

```
0055:       IF(J.EQ.1)GO TO 51
0056:       IF(J.EQ.N)GO TO 52
0057:       IF(I.GT.KXS1.AND.I.LT.KXS2)GO TO 53
0058:       IF(I.GT.LXS1.AND.I.LT.LXS2)GO TO 54
0059: 56    TY(I,J)=(T(I,J+1)-T(I,J-1))/DH*0.5
0060:       GO TO 50
0061: 51    TY(I,J)=0.
0062:       GO TO 50
0063: 52    TY(I,J)=TA(I)
0064:       GO TO 50
0065: 53    IF(J.GT.KYS)GO TO 56
0066:       IF(J.EQ.KYS)GO TO 57
0067:       TY(I,J)=0.
0068:       GO TO 50
0069: 57    TY(I,J)=0.
0070:       GO TO 50
0071: 54    IF(J.GT.LYS)GO TO 56
0072:       IF(J.EQ.LYS)GO TO 58
0073:       TY(I,J)=0.
0074:       GO TO 50
0075: 58    TY(I,J)=0.
0076: 50    CONTINUE
0077:       WRITE(IO,4)
0078: 4     FORMAT('1','THE VALUES OF D(P2)/DY'///)
0079:       CALL CREATE(100,'P2DYFILE')
0080:       DO 60 I=1,M
0081:       WRITE(100)(TY(I,J),J=1,N)
0082: 60    CONTINUE
0083:       WRITE(IO,2)((TY(I,J),J=1,N,IZ),I=1,M,IZ)
0084:       M1=M-1
0085:       DO 70 J=1,N
0086:       TY(1,J)=0.5*(4.*T(2,J)-3.*T(1,J)-T(3,J))/DH
0087:       TY(M,J)=0.5*(3.*T(M,J)-4.*T(M1 ,J)+T(M-2,J))/DH
0088:       DO 70 I=2,M1
0089:       IF(I.GE.KXS1.AND.I.LE.KXS2)GO TO 71
0090:       IF(I.GE.LXS1.AND.I.LE.LXS2)GO TO 72
0091: 73    TY(I,J)=0.5*(T(I+1,J)-T(I-1,J))/DH
0092:       GO TO 70
0093: 71    IF(J.GE.KYS)GO TO 73
0094:       TY(I,J)=0.
0095:       GO TO 70
0096: 72    IF(J.GE.LYS)GO TO 73
0097:       TY(I,J)=0.
0098: 70    CONTINUE
0099:       CALL CREATE(103,'P2DXFILE')
0100:       DO 80 I=1,M
0101:       WRITE(103)(TY(I,J),J=1,N)
0102: 80    CONTINUE
0103:       WRITE(IO,3)
0104: 3     FORMAT('1    THE VALUES OF D(P2)/DX'///)
0105:       WRITE(IO,2)((TY(I,J),J=1,N,IZ),I=1,M,IZ)
0106:       CALL EXIT
0107:       END
```

```
0001: C
0002: C     THIS PROGRAM CALCULATES THE ADVECTION TERM IN THE THETA 2 EQUATION
0003: C     SEE TEMPERATURE ZEROTH ORDER PROGRAM FOR INFORMATION ABOUT
0004: C     WESTINGHOUSE FMS OPEN AND CREATE ROUTINES
0005: C
0006:       DIMENSION CON(41,11),FA(41,11),FB(41,11)
0007:       DATA M/41/,N/11/,IN/2/,IO/3/
0008: 1     FORMAT(' ',3X,11F11.3)
0009:       ITER=0
0010:       DISCH=500.
0011:       M1=M-1
0012:       N1=N-1
0013:       DH=1./FLOAT(N1)
0014:       CALL OPEN(100,'P1DXFILE')
0015:       CALL OPEN(101,'T1DXFILE')
0016:       DO 20 I=1,M
0017:       READ(100)(FA(I,J),J=1,N)
0018:       READ(101)(FB(I,J),J=1,N)
0019: 20    CONTINUE
0020:       DO 500 I=1,M
0021:       DO 500 J=1,N
0022:       CON(I,J)=FA(I,J)*FB(I,J)
0023: 500   CONTINUE
0024:       CALL OPEN (100,'P1DYFILE')
0025:       CALL OPEN(101,'TEMOFILE')
0026:       DO 30 I=1,M
0027:       READ(100)(FA(I,J),J=1,N)
0028:       READ(101)(FB(I,J),J=1,N)
0029: 30    CONTINUE
0030:       DO 35 I=1,M
0031:       DO 35 J=1,N
0032:       FA(I,J)=FA(I,J)-FB(I,J)
0033: 35    CONTINUE
0034:       CALL OPEN (100,'T1DYFILE')
0035:       DO 40 I=1,M
0036:       READ(100)(FB(I,J),J=1,N)
0037: 40    CONTINUE
0038:       DO 50 I=1,M
0039:       DO 50 J=1,N
0040:       CON(I,J)=CON(I,J)+FA(I,J)*FB(I,J)
0041: 50    CONTINUE
0042:       CALL OPEN(100,'P2DXFILE')
0043:       CALL OPEN(101,'T0DXFILE')
0044:       DO 60 I=1,M
0045:       READ(100)(FA(I,J),J=1,N)
0046:       READ(101)(FB(I,J),J=1,N)
0047: 60    CONTINUE
0048:       DO 70 I=1,M
0049:       DO 70 J=1,N
0050:       CON(I,J)=CON(I,J)+FA(I,J)*FB(I,J)
0051: 70    CONTINUE
0052:       CALL OPEN(100,'P2DYFILE')
0053:       CALL OPEN(101,'TEM1FILE')
0054:       DO 80 I=1,M
```

```
0055:        READ(100)(FA(I,J),J=1,N)
0056:        READ(101)(FB(I,J),J=1,N)
0057: 80     CONTINUE
0058:        DO 90 I=1,M
0059:        DO 90 J=1,N
0060:        FA(I,J)=FA(I,J)-FB(I,J)
0061: 90     CONTINUE
0062:        CALL OPEN(100,'TODYFILE')
0063:        DO 100 I=1,M
0064:        READ(100)(FB(I,J),J=1,N)
0065: 100    CONTINUE
0066:        DO 110 I=1,M
0067:        DO 110 J=1,N
0068:        CON(I,J)=CON(I,J)+FA(I,J)*FB(I,J)
0069:        CON(I,J)=CON(I,J)*DISCH*DH*DH
0070: 110    CONTINUE
0071:        WRITE(10,13)
0072: 13     FORMAT('1','   THE TEMPERATURE 2 ADVECTION TERM'///)
0073:        WRITE(3,502) DISCH
0074: 502    FORMAT('0', 'DISCHARGE = ',F6.0)
0075:        CALL CREATE(100,'TEM2TERM')
0076:        WRITE(10,1)((CON(I,J),J=1,N),I=1,M)
0077:        DO 610 I=1,M
0078:        WRITE(100)(CON(I,J),J=1,N)
0079: 610    CONTINUE
0080:        STOP
0081:        END
```

```
0001: C
0002: C      THIS PROGRAM ITERATES THE TEMPERATURE SECOND ORDER SOLUTION
0003: C      WITH VERTICAL HEAT SOURCES
0004: C      SEE TEMPERATURE ZEROTH ORDER PROGRAM FOR INFORMATION ABOUT
0005: C      WESTINGHOUSE FMS OPEN AND CREATE ROUTINES
0006: C
0007:        DIMENSION FA(41,11),FB(41,11),CON(41,11)
0008:        DIMENSION E1(41),E2(41)
0009:        DATA FB/451*0.0/
0010: 1      FORMAT(4X,11F11.3)
0011: 3      FORMAT(3F5.0)
0012:        DERR=0.0001
0013:        IIN=2
0014:        IO=3
0015:        M=41
0016:        N=11
0017:        M1=M-1
0018:        N1=N-1
0019:        DH=1./FLOAT(N1)
0020:        READ(IIN,3)YS,XS1,XS2
0021:        READ(IIN,3)YS2,XS21,XS22
0022:        LYS=IFIX(YS2/DH+0.1)+1
0023:        LXS1=IFIX(XS21/DH+0.1)+1
0024:        LXS2=IFIX(XS22/DH+0.1)+1
0025:        KYS=IFIX(YS/DH+0.1)+1
0026:        KXS1=IFIX(XS1/DH+0.1)+1
0027:        KXS2=IFIX(XS2/DH+0.1)+1
0028:        WRITE(IOUT,4)YS,XS1,XS2
0029:        WRITE(IOUT,4)YS2,XS21,XS22
0030: 4      FORMAT('1THE VERTICAL SOURCE IS AT:','Y <',F5.1,',',F5.1,'<X<',
0031:       1  F5.1)
0032:        CALL OPEN(103,'T1DYFILE')
0033:        CALL OPEN(104,'PRS1FILE')
0034:        DO 20 I=1,M
0035:        READ(103) (FA(I,J),J=1,N)
0036:        READ(104) (CON(I,J),J=1,N)
0037:        E1(I)=FA(I,N)
0038:   20   FB(I,N)= -(FA(I,N) * CON(I,N))
0039:        CALL OPEN(103,'P1DYFILE')
0040:        DO 21 I=1,M
0041:        READ(103)(FA(I,J),J=1,N)
0042:        E2(I)=FA(I,N)*CON(I,N)
0043: 21     CONTINUE
0044:        CALL OPEN(103,'PRS2FILE')
0045:        DO 22 I=1,M
0046:        READ(103)(FA(I,J),J=1,N)
0047:        E2(I)=E2(I)+FA(I,N)
0048: 22     CONTINUE
0049:        CALL OPEN(103,'T0DYFILE')
0050:        DO 23 I=1,M
0051:        READ(103)(FA(I,J),J=1,N)
0052:        FB(I,N)=FB(I,N)-E2(I)*FA(I,N)-E1(I)*E1(I)*(FA(I,N)-FA(I,N-1))/DH
0053: 23     CONTINUE
0054:        CALL OPEN(100,'TEM2TERM')
```

```
0055:        DO 13 I=1,M
0056: 13     READ(100)(CON(I,J),J=1,N)
0057: 900    DMAX=0.
0058:        FMAX=0.
0059:        DO 700 I=2,M1
0060:        DO 700 J=2,N1
0061:        IF(I.GE.KXS1.AND.I.LE.KXS2.AND.J.LE.KYS)GO TO 18
0062:        IF(I.GE.LXS1.AND.I.LE.LXS2.AND.J.LE.LYS)GO TO 18
0063:        GO TO 19
0064: 18     FB(I,J)=0.0
0065:         GO TO 700
0066: 19     TEMP=FB(I,J)
0067:        FB(I,J)=FB(I+1,J)+FB(I-1,J)+FB(I,J+1)+FB(I,J-1)+CON(I,J)
0068:        FB(I,J)=0.25*FB(I,J)
0069:        DAB=ABS(FB(I,J)-TEMP)
0070:        FAB=ABS(FB(I,J))
0071:        IF(DAB.GT.DMAX)DMAX=DAB
0072:        IF(FAB.GT.FMAX)FMAX=FAB
0073: 700    CONTINUE
0074:        ITER=ITER+1
0075:        DAB=DMAX/FMAX
0076:        IF(DAB.GT.DERR)GO TO 900
0077: 11     FORMAT('1     THE T2 FUNCTION'///)
0078: 12     FORMAT('0','      AFTER',I5,'  ITERATIONS')
0079:        WRITE(10,11)
0080:        WRITE(10,12)ITER
0081:        WRITE(10,1)((FB(I,J),J=1,N),I=1,M)
0082:        CALL CREATE(100,'TEM2FILE')
0083:        DO 910 I=1,M
0084:        WRITE(100)(FB(I,J),J=1,N)
0085: 910    CONTINUE
0086:        STOP
0087:        END
```

```
0001: C
0002: C      THIS PROGRAM WILL COMPUTE THE SECOND-ORDER STREAM FUNCTION
0003: C      VALUES WITH DIKE INTRUSION
0004: C
0005:        DIMENSION P(41,11),  TX(41,11)
0006:        DATA P/451*0.0/
0007:        IN=2
0008:        IO=3
0009:        M=41
0010:        N=11
0011:        M1=M-1
0012:        N1=N-1
0013:        DH=1.0/FLOAT(N1)
0014:        ITER=0
0015:        DERR=0.0001
0016: C
0017: C      READ IN DIKES LOCATIONS
0018: C
0019:        READ(IN,2)YS,XS1,XS2
0020:        READ(IN,2)YS2,XS21,XS22
0021: 2      FORMAT(3F5.0)
0022:        LYS=IFIX(YS2/DH+0.1)+1
0023:        LXS1=IFIX(XS21/DH+0.1)+1
0024:        LXS2=IFIX(XS22/DH+0.1)+1
0025:        KYS=IFIX(YS/DH+0.1)+1
0026:        KXS1=IFIX(XS1/DH+0.1)+1
0027:        KXS2=IFIX(XS2/DH+0.1)+1
0028: C
0029: C      COMPUTE STREAM FUNCTION 2 (X,1) BOUNDARY CONDITION
0030: C
0031:        CALL OPEN(100,'PRS1FILE')
0032:        DO 200 I=1,M
0033:        READ(100)(TX(I,J),J=1,N)
0034:        P(I,N)=TX(I,N)
0035: 200    CONTINUE
0036:        CALL OPEN(100,'STREAMF1')
0037:        DO 210 I=1,M
0038:        READ(100)(TX(I,J),J=1,N)
0039:        P(I,N)=P(I,N)*(TX(I,N1)-TX(I,N))/DH
0040: 210    CONTINUE
0041: C
0042: C      READ IN D(T1)/DX VALUES
0043: C
0044:        CALL OPEN (100,'T1DXFILE')
0045:        DO 12 I=1,M
0046:        READ(100) (TX(I,J),J=1,N)
0047: 12     CONTINUE
0048: 15     DMAX=0.0
0049:        FMAX=0.0
0050:        DO 20 I=1,M
0051:        DO 20 J=2,N1
0052:        TEMP=P(I,J)
0053:        IF(I.GE.KXS1.AND.I.LE.KXS2)GO TO 23
0054:        IF(I.GE.LXS1.AND.I.LE.LXS2)GO TO 24
```

```
0055:        GO TO 22
0056: 23     IF(J.GT.KYS)GO TO 22
0057:        GO TO 20
0058: 24     IF(J.GT.LYS)GO TO 22
0059:        GO TO 20
0060: 22     IF(I.EQ.1) GO TO 25
0061:        IF(I.EQ.M) GO TO 26
0062:        P(I,J)=P(I+1,J) + P(I-1,J) + P(I,J-1) + P(I,J+1) + DH*DH*TX(I,J)
0063:        P(I,J)=0.25*P(I,J)
0064:        GO TO 30
0065: 25     P(1,J)=P(1,J-1) + P(1,J+1) + 2.*P(2,J) + DH*DH*TX(1,J)
0066:        P(1,J)=P(1,J)*0.25
0067:        GO TO 30
0068: 26     P(M,J) = P(M,J-1) + P(M,J+1) + 2.*P(M1,J) + DH*DH*TX(M,J)
0069:        P(M,J) = P(M,J)*0.25
0070: 30     DABV=ABS(P(I,J)-TEMP)
0071:        DFAB=ABS(P(I,J))
0072:        IF(DABV.GT.DMAX) DMAX=DABV
0073:        IF(DFAB.GT.FMAX) FMAX=DFAB
0074: 20     CONTINUE
0075:        DABV=DMAX/FMAX
0076:        ITER=ITER+1
0077:        IF(DABV.GT.DERR) GO TO 15
0078:        WRITE(IO,5)
0079:        WRITE(IO,8)
0080: 5      FORMAT('1', ' THE STREAM FUNCTION SECOND-ORDER VALUES')
0081: 8      FORMAT('      X 10(-2)'///)
0082:        CALL CREATE(102,'STREAMF2')
0083:        DO 100 I=1,M
0084:        WRITE(102) (P(I,J),J=1,N)
0085: 100    CONTINUE
0086:        DO 50 I=1,M
0087:        DO 51 J=1,N
0088: 51     P(I,J)=P(I,J)*100.
0089: 50     WRITE(IO,6)(P(I,J),J=1,N)
0090: 6      FORMAT(' ',3X,11F11.3)
0091:        WRITE(IO,7) ITER
0092: 7      FORMAT(//// 4X,'ITERATION CONVERGES AFTER',I4,' TRIES')
0093:        STOP
0094:        END
```

```
0001: C
0002: C      THIS PROGRAM ITERATES THE STREAM FUNCTION 1
0003: C      WITH VERTICAL HEAT SOURCES
0004: C      SEE TEMPERATURE ZEROTH ORDER PROGRAM FOR INFORMATION ABOUT
0005: C      WESTINGHOUSE FMS OPEN AND CREATE ROUTINES
0006: C
0007:        DIMENSION P(41,11),  TX(41,11)
0008:        DATA P/451*0.0/
0009:        IN=2
0010:        IO=3
0011:        M=41
0012:        N=11
0013:        M1=M-1
0014:        N1=N-1
0015:        DH=1.0/FLOAT(N1)
0016:        ITER=0
0017:        DERR=0.00001
0018:        READ(IN,2)YS,XS1,XS2
0019:        READ(IN,2)YS2,XS21,XS22
0020: 2      FORMAT(3F5.0)
0021:        LYS=IFIX(YS2/DH+0.1)+1
0022:        LXS1=IFIX(XS21/DH+0.1)+1
0023:        LXS2=IFIX(XS22/DH+0.1)+1
0024:        KYS=IFIX(YS/DH+0.1)+1
0025:        KXS1=IFIX(XS1/DH+0.1)+1
0026:        KXS2=IFIX(XS2/DH+0.1)+1
0027:        CALL OPEN(100,'TODXFILE')
0028:        DO 12 I=1,M
0029:        READ(100) (TX(I,J),J=1,N)
0030: 12     CONTINUE
0031: 15     DMAX=0.0
0032:        FMAX=0.0
0033:        DO 20 I=1,M
0034:        DO 20 J=2,N1
0035:        TEMP=P(I,J)
0036:        IF(I.GE.KXS1.AND.I.LE.KXS2)GO TO 23
0037:        IF(I.GE.LXS1.AND.I.LE.LXS2)GO TO 24
0038:        GO TO 22
0039: 23     IF(J.GT.KYS)GO TO 22
0040:        GO TO 20
0041: 24     IF(J.GT.LYS)GO TO 22
0042:        GO TO 20
0043: 22     IF(I.EQ.1) GO TO 25
0044:        IF(I.EQ.M) GO TO 26
0045:        P(I,J)=P(I+1,J) + P(I-1,J) + P(I,J-1) + P(I,J+1) + DH*DH*TX(I,J)
0046:        P(I,J)=0.25*P(I,J)
0047:        GO TO 30
0048: 25     P(1,J)=P(1,J-1) + P(1,J+1) + 2.*P(2,J) + DH*DH*TX(1,J)
0049:        P(1,J)=P(1,J)*0.25
0050:        GO TO 30
0051: 26     P(M,J) = P(M,J-1) + P(M,J+1) + 2.*P(M1,J) + DH*DH*TX(M,J)
0052:        P(M,J) = P(M,J)*0.25
0053: 30     DABV=ABS(P(I,J)-TEMP)
0054:        DFAB=ABS(P(I,J))
```

```
0055:          IF(DABV.GT.DMAX) DMAX=DABV
0056:          IF(DFAB.GT.FMAX) FMAX=DFAB
0057: 20       CONTINUE
0058:          DABV=DMAX/FMAX
0059:          ITER=ITER+1
0060:          IF(DABV.GT.DERR) GO TO 15
0061:          WRITE(IO,5)
0062: 5        FORMAT('1   THE STREAM FUNCTION 1  (X 100)'///)
0063:          CALL CREATE(102,'STREAMF1')
0064:          DO 100 I=1,M
0065:          WRITE(102) (P(I,J),J=1,N)
0066: 100      CONTINUE
0067:          DO 50 I=1,M
0068:          DO 51 J=1,N
0069: 51       P(I,J)=P(I,J)*100.
0070: 50       WRITE(IO,6)(P(I,J),J=1,N)
0071: 6        FORMAT(' ',3X,11F11.3)
0072:          WRITE(IO,7) ITER
0073: 7        FORMAT(//// 4X,'ITERATION CONVERGES AFTER',I4,' TRIES')
0074:          STOP
0075:          END
```

```
0001:  C
0002:  C      THIS PROGRAM COMPUTES THE TOTAL SOLUTION FOR THE TEMPERATURE
0003:  C      FIELD SOLUTION
0004:  C
0005:        DIMENSION T(41,11),TA(41,11)
0006:        DATA IN/2/,IO/3/,ESL/0.1/,ESL2/0.01/,M/41/,N/11/
0007:  C
0008:  C      READ IN TO VALUES
0009:  C
0010:        CALL OPEN(100,'TEMOFILE')
0011:        DO 10 I=1,M
0012:        READ(100)(T(I,J),J=1,N)
0013:  10    CONTINUE
0014:        CALL OPEN(100,'TEM1FILE')
0015:        DO 20 I=1,M
0016:        READ(100)(TA(I,J),J=1,N)
0017:  20    CONTINUE
0018:        DO 30 I=1,M
0019:        DO 30 J=1,N
0020:        T(I,J)=T(I,J)+ESL*TA(I,J)
0021:  30    CONTINUE
0022:  C
0023:  C       READ IN T2 VALUES
0024:  C
0025:        CALL OPEN(100,'TEM2FILE')
0026:        DO 40 I=1,M
0027:        READ(100)(TA(I,J),J=1,N)
0028:  40    CONTINUE
0029:        CALL CREATE(100,'TEMPFILE')
0030:        DO 50 I=1,M
0031:        DO 50 J=1,N
0032:        T(I,J)=T(I,J)+ESL2*TA(I,J)
0033:  50    CONTINUE
0034:        WRITE(IO,110)
0035:        DO 60 I=1,M
0036:        WRITE(IO,100)(T(I,J),J=1,N)
0037:        WRITE(100)(T(I,J),J=1,N)
0038:  60    CONTINUE
0039:  100   FORMAT(1X,11F12.4)
0040:  110   FORMAT('1',' THE TEMPERTURE T = TO + E*T1 + E2*T2'///)
0041:        STOP
0042:        END
```

```
001: C
002: C      THE PROGRAM TO COMPUTE P0 + E*P1 +E2*P2
003: C
004:        DIMENSION P1(41,11),P(41,11)
005:        CALL OPEN(100,'PRS1FILE')
006:        M=41
007:        N=11
008:        DH=1.0/FLOAT(N-1)
009:        DO 10 I=1,41
010:        READ(100)(P1(I,J),J=1,11)
011: 10     CONTINUE
012: 1      FORMAT(' ',3X,11F11.3)
013: 2      FORMAT('1','    THE PRESURE FUNCTION')
014:        Y=0.
015:        DO 15 J=1,11
016:        DO 20 I=1,41
017:        P(I,J)=1.0-Y
018: 20     CONTINUE
019:        Y=Y+DH
020: 15     CONTINUE
021:        DO 30 I=1,M
022:        DO 30 J=1,N
023:        P(I,J)=P(I,J)+P1(I,J)*0.1
024: 30     CONTINUE
025:        CALL OPEN(100,'PRS2FILE')
026:        DO 60 I=1,M
027:        READ(100)(P1(I,J),J=1,N)
028: 60     CONTINUE
029:        DO 70 I=1,M
030:        DO 70 J=1,N
031:        P(I,J)=P(I,J)+0.01*P1(I,J)
032: 70     CONTINUE
033:        CALL CREATE(102,'PRESSURE')
034:        WRITE(3,2)
035:        DO 50 I=1,M
036:        WRITE(102)(P(I,J),J=1,N)
037:        WRITE(3,1)(P(I,J),J=1,N)
038: 50     CONTINUE
039:        STOP
040:        END
```

```
0001: C
0002: C      PROGRAM TO COMPUTE STREAM FUNCTION
0003: C      S = S1 * E1 + S2 * E2
0004: C
0005:        DIMENSION S1(41,11),S(41,11)
0006:        CALL OPEN(100,'STREAMF1')
0007:        M=41
0008:        N=11
0009:        DH=1.0/FLOAT(N-1)
0010:        DO 10 I=1,41
0011:        READ(100)(S1(I,J),J=1,N)
0012: 10     CONTINUE
0013: 1      FORMAT(' ',3X,11F11.3)
0014: 2      FORMAT('1','    THE STREAM FUNCTION '///)
0015:        DO 20 I=1,M
0016:        DO 20 J=1,N
0017:        S(I,J)=0.1*S1(I,J)
0018: 20     CONTINUE
0019:        CALL OPEN(100,'STREAMF2')
0020:        DO 30 I=1,M
0021:        READ(100)(S1(I,J),J=1,N)
0022: 30     CONTINUE
0023:        DO 40 I=1,M
0024:        DO 40 J=1,N
0025:        S(I,J)=S(I,J)+0.01*S1(I,J)
0026: 40     CONTINUE
0027:        WRITE(3,2)
0028:        CALL CREATE(102,'STREAMFN')
0029:        DO 50 I=1,M
0030:        WRITE(102)(S(I,J),J=1,N)
0031:        WRITE(3,1)(S(I,J),J=1,N)
0032: 50     CONTINUE
0033:        STOP
0034:        END
```